

USER GUIDE

Essential Studio for EJ2 React

Version - v25.1.35 | Release Date - March 15, 2024

Calendar	28
Getting started	28
Dependencies.....	28
Installation and configuration.....	28
Adding Syncfusion packages	28
Adding Style sheet to the Application.....	29
Adding Calendar component to the Application	29
Run the application	30
Setting the value, min and max dates.....	31
See Also	32
Date range in React Calendar component.....	32
Globalization in React Calendar component	34
Right-To-Left	551
Customization in React Calendar component	1130
Disable Weekends.....	1131
Day Cell Format.....	1132
Highlight Weekends	1136
See Also	1138
Multi select in React Calendar component.....	1138
Calendar views in React Calendar component	1139
View Restriction	1140
Accessibility in React Calendar component.....	1143
WAI-ARIA attributes.....	1144
Keyboard Interaction	1144
Ensuring accessibility	1147
See also	1147
Islamic calendar in React Calendar component.....	1147
Style appearance in React Calendar component.....	1151
Customizing the background color for the Calendar	1151
Customizing the Calendar date elements on hovering.....	1152
Customizing the border of date cell grid	1152
Customizing the Calendar title.....	1152
Customizing the previous and next icon.....	1152
Customizing the footer button	1153
Customizing the selected date cell grid	1153

Customizing the content header in Calendar	1153
How To	1153
Set clear button in calendar in React Calendar component	1153
Show dates of other months in React Calendar component.....	1155
Select a sequence of dates in calendar in React Calendar component.....	1156
Skip a month in calendar in React Calendar component.....	1158
Render the calendar with week numbers in React Calendar component.....	1160
Change the first day of week in React Calendar component.....	1160
Customize the calendar day header in React Calendar component.....	1161
Card.....	1163
Getting Started.....	1163
Dependencies.....	1163
Setup for Local Development.....	1163
Adding Syncfusion packages	1164
Adding a simple Card	1164
Adding a header to the card	1164
See Also	1168
Header content in React Card component	1168
Header.....	1168
Content	1172
Card image in React Card component	1176
Images	1176
Divider	1179
See Also	1183
Action buttons in React Card component.....	1183
Vertical	1183
See Also	1188
Horizontal in React Card component.....	1188
Stacked cards	1188
Style in React Card component.....	1192
Customizing the card	1192
Customizing the Header element	1192
Customizing the card content.....	1192
Divider used to separate the elements inside the card.....	1193
Including image within card element	1193

Including a title or caption for the image	1193
To include heading image within the header	1193
Customizing the Header main title	1194
Customizing the Header subtitle.....	1194
Including action buttons or anchor tags	1194
To align card elements horizontally	1194
To align elements vertically within the horizontal layout.....	1195
How To	1195
Customize the card image title position in React Card component	1195
Integrate other component inside the card in React Card component.....	1200
Carousel	1202
Getting Started.....	1202
Dependencies.....	1202
Setup your development environment	1203
Adding Syncfusion packages	1203
Adding Style sheet to the Application.....	1204
Add Carousel to the project	1204
Run the application	1205
Populating items in React Carousel component.....	1205
Populating items using carousel item	1205
Populating Items using data source	1207
Selection.....	1208
Partial visible slides	1212
See Also	1215
Navigators and indicators in React Carousel component.....	1215
Navigators	1216
Indicators	1221
Play button.....	1235
Animations and transitions in React Carousel component	1239
Animations	1239
Intervals between slides	1242
Auto play slides	1244
Pause on hover.....	1245
Looping slides.....	1247
Slide changing events.....	1249

Disable touch swiping	1251
Swipe Modes.....	1252
Accessibility in React Carousel component	1254
ARIA attributes.....	1255
Keyboard interaction	1256
Ensuring accessibility	1256
See also	1256
Styles and appearance in React Carousel component.....	1256
CSS Structure in React Carousel Control.....	1257
Customizing the indicators	1257
Customizing the navigators.....	1259
Customizing partial slides size	1261
Chart.....	1261
Getting Started.....	1261
Getting Started.....	1261
Creating a Next.js Application Using Syncfusion React Components	1273
Getting Started with the React Chart Component in the Preact Framework.....	1275
Working with data in React Chart component	1279
Local Data.....	1279
Lazy loading.....	1280
Remote Data	1284
Empty points	1286
Chart dimensions in React Chart component	1288
Size for Container.....	1288
Size for Chart.....	1290
Category axis in React Chart component.....	1293
Labels Placement	1294
Range	1295
Indexed category axis.....	1296
Numeric axis in React Chart component	1297
Range	1299
Range Padding.....	1300
Label Format	1306
GroupingSeparator	1308
Custom Label Format.....	1309

Date time axis in React Chart component	1310
DateTime Axis	1310
DateTimeCategory Axis	1312
Label Format	1319
Logarithmic axis in React Chart component	1321
Range	1322
Logarithmic Base	1324
Logarithmic Interval	1325
Axis labels in React Chart component	1326
Smart Axis Labels	1326
Axis Labels Positioning	1330
Multilevel Labels	1331
Sorting	1339
Edge Label Placement	1340
Labels Customization	1341
Customizing Specific Point	1342
Trim using maximum label width	1344
Line break support	1345
Axis customization in React Chart component	1347
Axis Crossing	1347
Title	1348
Title Rotation	1349
Tick Lines Customization	1350
Grid Lines Customization	1352
Multiple Axis	1354
Inversed Axis	1356
Opposed Position	1357
Strip line in React Chart component	1358
Horizontal Strip lines	1358
Vertical Striplines	1360
Customize the strip line	1361
Customize the stripline text	1362
Dash Array	1364
Recurrence Stripline	1365
Size Type	1366

Segment Stripline.....	1368
See Also	1369
Multiple panes in React Chart component	1369
Rows.....	1370
Columns	1374
Chart Types	1378
Line Chart in React Chart component.....	1378
Step line in React Chart component	1382
Stacked Line in React Chart component	1384
100% Stacked Line in React Chart component	1387
Spline in React Chart component	1391
Area series in React Chart component	1393
Range Area in React Chart component.....	1399
Range step area Chart in React Chart component	1401
Spline Range Area in React Chart component	1404
Stacked Area in React Chart component	1408
100% Stacked Area in React Chart component	1412
Stacked step area Chart in React Chart component.....	1415
Step area in React Chart component.....	1418
Step Area in React Chart component.....	1420
Column Chart in React Chart component	1422
Range Column Chart in React Chart component	1433
Stacked Column Chart in React Chart component	1437
100% Stacked Column Chart in React Chart component.....	1442
Bar Charts in React Chart component	1447
Stacked Bar Charts in React Chart component.....	1458
100% Stacked Bar Charts in React Chart component	1464
Bubble in React Chart component.....	1468
Scatter in React Chart component.....	1472
Polar Charts in React Chart component.....	1476
Radar Charts in React Chart component	1487
Hilo Charts in React Chart component.....	1491
High Low Open Close in React Chart component	1494
Candle in React Chart component	1497
Box and Whisker in React Chart component	1501

Waterfall in React Chart component	1505
Histogram in React Chart component	1509
Error Bar in React Chart component.....	1512
Vertical Chart in React Chart component	1521
Pareto in React Chart component	1523
Chart series in React Chart component	1526
Multiple Series	1526
Combination Series	1528
Technical indicators in React Chart component	1530
Accumulation Distribution	1530
Average True Range (ATR)	1533
Bollinger Band.....	1535
Exponential Moving Average (EMA)	1539
Momentum	1541
Moving Average Convergence Divergence (MACD)	1545
Relative Strength Index (RSI).....	1550
Simple Moving Average (SMA)	1552
Stochastic	1554
Triangular Moving Average (TMA).....	1559
Trend lines in React Chart component	1565
Linear.....	1565
Exponential	1568
Logarithmic	1570
Polynomial	1572
Power	1574
Moving Average	1576
Forecasting.....	1581
Show or hide a trendline.....	1585
Data markers in React Chart component	1587
Marker.....	1587
Shape.....	1589
Images.....	1590
Customization	1592
Customizing specific point	1593
Fill marker with series color	1595

See Also	1596
Data labels in React Chart component	1596
Position	1598
Data Label Template	1599
Text Mapping	1601
Format.....	1602
Margin	1603
DataLabel Rotation	1605
Customization	1606
Customizing Specific Point	1608
Show percentage based on each series points.....	1609
See Also	1612
Chart annotations in React Chart component	1612
Region	1613
Co-ordinate Units.....	1615
Legend in React Chart component.....	1617
Position and Alignment.....	1617
Legend Reverse	1620
Customization	1625
Series Selection on Legend	1638
Enable Animation.....	1640
Collapsing Legend Item	1642
Legend Title.....	1643
Arrow Page Navigation	1645
Legend Item Padding	1648
See Also	1650
Tooltip in React Chart component.....	1650
Default tooltip.....	1650
Fixed tooltip	1652
Format the tooltip.....	1653
Individual series format	1654
Tooltip template	1657
Tooltip mapping name.....	1659
Customize the appearance of tooltip	1660
See also	1661

Zooming in React Chart component	1662
Enable zooming.....	1662
Modes	1663
Toolbar	1665
Enable scrollbar.....	1666
Enable pan.....	1668
Auto interval on zooming.....	1670
Data editing in React Chart component.....	1671
Enable Data Editing	1671
Cross hair and track ball in React Chart component.....	1673
Tooltip for axis	1675
Customization	1676
Trackball.....	1678
Synchronized Charts in React Chart component	1680
Tooltip synchronization.....	1680
Crosshair synchronization	1685
Zooming synchronization.....	1690
Selection synchronization	1695
Selection in React Chart component	1701
Point.....	1701
Series.....	1702
Cluster	1704
Rectangular selection.....	1705
Lasso selection	1706
Multi-region selection.....	1708
Selection Type.....	1709
Selection on Load	1710
Selection through on legend.....	1712
Customization for selection	1714
See Also	1715
Chart print in React Chart component.....	1715
Print.....	1715
Export.....	1717
Multiple chart export.....	1728
Exporting chart using base64 string.....	1731

Chart appearance in React Chart component	1734
Custom color palette.....	1734
Data point customization.....	1735
Point level customization.....	1739
Chart area customization.....	1741
Animation.....	1746
Chart title	1750
Chart subTitle	1759
See also	1761
Render methods in React Chart component	1761
SVG.....	1761
Canvas	1762
Accessibility in React Chart component.....	1762
WAI-ARIA attributes.....	1763
Keyboard interaction	1763
Ensuring accessibility	1764
See also	1764
Internationalization in React Chart component	1764
Localization in React Chart component	1766
Ej1 api migration in React Chart component.....	1768
Annotations.....	1768
Columns	1770
CommonSeriesOptions	1770
Crosshair	1771
3D chart.....	1771
Indicators	1771
Legend.....	1774
primaryXAxis	1776
primaryYAxis	1782
Axes.....	1789
Rows.....	1796
Series.....	1796
marker.....	1801
TrendLines.....	1804
StripLines.....	1805

Multilevel Labels	1807
Methods	1808
Events.....	1809
Chart properties.....	1813
How To	1814
Live chart in React Chart component.....	1814
Prevent data label in React Chart component.....	1816
Tool tip format in React Chart component	1818
Add series in React Chart component	1820
Points customization in React Chart component	1822
Stacking total in React Chart component	1824
Selected data grid in React Chart component	1827
Marker customization in React Chart component.....	1831
Legend customization in React Chart component.....	1833
Tool tip table in React Chart component.....	1835
Footer in React Chart component.....	1838
Threshold in React Chart component	1840
Grid data chart in React Chart component.....	1841
Data label template in React Chart component	2318
Check box.....	2320
Getting Started.....	2320
Dependencies.....	2320
Installation and Configuration	2321
Adding Syncfusion packages	2321
Adding CSS Reference	2321
Adding CheckBox component to the Application	2322
Run the application	2322
States in React Check box component.....	2323
Checked and Unchecked	2323
Indeterminate	2323
Label and size in React Check box component	2324
Label	2324
Size	2325
See Also	2326
Style and appearance in React Check box component.....	2326

Accessibility in React CheckBox component.....	2326
WAI-ARIA attributes.....	2327
Keyboard interaction	2327
Ensuring accessibility	2327
See also	2328
How To	2328
Customized checkbox in React Check box component.....	2328
Name and value in form submit in React Check box component.....	2331
Right to left in React Check box component.....	2332
Ej1 api migration in React Check box component	2333
Properties.....	2333
Methods.....	2334
Events.....	2335
Chips.....	2335
Getting Started.....	2335
Dependencies.....	2335
Installation and Configuration	2336
Adding Syncfusion packages	2336
Adding CSS Reference	2336
Adding Chip component to the Application	2336
Running the application	2337
Types in React Chips component	2338
Input Chip.....	2338
Choice Chip	2339
Filter Chip	2340
Action Chip	2341
Customization in React Chips component	2343
Styles	2343
Leading Icon	2344
Avatar.....	2345
Avatar Content.....	2346
Trailing Icon.....	2347
Outline Chip	2348
Style in React Chips component	2349
Customizing the chip text	2349

Customizing the chip icon	2349
Customizing the chip delete button.....	2350
Customizing the chip outline	2350
Customizing the chip on selection	2350
Customizing the chip avatar text	2351
Accessibility in React Chips component.....	2351
WAI-ARIA attributes.....	2352
Keyboard interaction	2352
Ensuring accessibility	2352
See also	2353
3D Circular Chart.....	2353
Getting started	2353
Dependencies.....	2353
Installation and configuration.....	2353
Add 3D Circular Chart to the project	2354
Pie and Donut in React 3D Circular Chart component.....	2356
Pie chart	2356
Radius customization	2358
Various radius pie chart	2359
Donut chart	2361
Text and fill color mapping	2363
Customization	2365
Data Label in React 3D Circular Chart component	2367
Positioning.....	2369
Data label template.....	2371
Connector line.....	2373
Text mapping	2375
Format.....	2377
Customization	2379
Using textRender event	2381
Using template.....	2383
Empty points in React 3D Circular Chart component	2385
Customization	2386
Legend in React 3D Circular Chart component.....	2388
Position and alignment	2390

Legend reverse	2392
Legend shape	2393
Legend size	2395
Legend item size.....	2397
Legend paging	2399
Legend text wrap	2400
Legend title	2402
Arrow page navigation.....	2404
Legend item padding	2406
Tooltip in React 3D Circular Chart component	2408
Header.....	2410
Format.....	2412
Tooltip template	2414
Fixed tooltip	2416
Customization	2418
Customization of individual tooltip.....	2420
Title and subtitle in React 3D Circular Chart component	2422
Title	2422
Title customization.....	2424
Subtitle	2426
Subtitle customization	2428
Print and Export in React 3D Circular Chart component	2430
Print.....	2430
Export.....	2433
Circular Gauge.....	2436
Getting Started.....	2436
Dependencies.....	2436
Installation and configuration.....	2437
Set Pointer Value	2439
Creating a Next.js Application Using Syncfusion React Components	2440
What is Next.js?	2440
Prerequisites	2440
Create a Next.js application	2440
Install Syncfusion React packages.....	2441
Add Syncfusion React component	2441

Run the application	2441
Gauge dimensions in React Circular gauge component	2442
Size for Container	2442
Size for Circular Gauge	2442
Gauge axes in React Circular Gauge component	2444
Axis Customization	2444
Angles and Direction	2445
Axis Radius	2447
Ticks.....	2448
Labels	2451
Minimum and Maximum	2459
Multiple Axes	2459
Gauge ranges in React Circular gauge component	2461
Start and End.....	2461
Customization	2462
Radius.....	2463
Dragging range	2465
Multiple Ranges	2466
Rounded corner radius	2467
Gradient Color.....	2468
See also	2484
Gauge pointers in React Circular gauge component	2484
Needle Pointers.....	2485
RangeBar Pointer	2490
Rounded corner for range bar pointer	2492
Marker Pointer	2493
Dragging pointer	2495
Multiple Pointers	2496
Animation.....	2498
Gradient Color.....	2499
Gauge annotations in React Circular gauge component	2505
Content	2505
Position	2506
Sub Gauge	2507
See also	2512

Animation in React Circular Gauge component	2512
Gauge legend in React Circular gauge component.....	2516
Legend customization	2516
Toggle option in legend	2519
Paging support in legend	2520
Legend text customization.....	2522
Gauge user interaction in React Circular gauge component	2523
Tooltip for pointers	2523
Tooltip for ranges.....	2524
Tooltip for annotation.....	2524
Pointer Drag	2528
Gauge print and export in React Circular gauge component	2529
Print.....	2529
Export.....	2530
Gauge appearance in React Circular gauge component.....	2533
Gauge Title	2533
Gauge Position	2534
Area Customization.....	2536
Radius calculation based on angles	2539
Accessibility in React Circular Gauge component.....	2540
WAI-ARIA attributes.....	2540
Screen reading in Circular Gauge.....	2540
Ensuring accessibility	2541
See also	2541
Internationalization in React Circular Gauge component	2541
Globalization	2541
Right-to-left.....	2542
Ej1 api migration in React Circular gauge component.....	2546
Circular gauge dimensions	2546
Axis Line	2546
Ticks.....	2548
Labels	2550
Ranges.....	2552
Needle Pointer	2554
Marker Pointer	2555

Rangebar Pointer	2556
Annotations.....	2557
Appearance	2558
Events.....	2559
ColorPicker	2561
Getting Started.....	2561
Dependencies.....	2561
Installation and configuration.....	2561
Adding Syncfusion packages	2562
Adding CSS reference.....	2562
Adding ColorPicker to the application	2562
Run the application	2563
See Also	2564
Mode and value in React Color picker component	2564
Inline	2564
Rendering palette at initial load	2565
Color value	2566
See Also	2567
Localization in React Color picker component.....	2567
Localization	2567
Right to Left - RTL.....	2568
See Also	2569
Accessibility in React ColorPicker component.....	2569
WAI-ARIA attributes.....	2570
Keyboard interaction	2571
Ensuring accessibility	2571
See also	2571
Style and appearance in React Color picker component.....	2571
How To	2572
Hide control buttons in React Color picker component	2572
Render palette alone in React Color picker component.....	2572
Colorpicker in dropdownbutton in React Color picker component	2573
Customize colorpicker in React Color picker component.....	2575
Handle no color support in React Color picker component	2585
Disabled in React Color picker component.....	2591

Ej1 api migration in React Color picker component	2592
Properties.....	2592
Methods	2594
Events.....	2596
ComboBox.....	2597
Getting Started.....	2597
Dependencies.....	2597
Installation and configuration.....	2597
Adding syncfusion packages	2598
Adding ComboBox component	2598
Adding CSS reference.....	2599
Binding data source	2599
Run the application	2600
Custom values.....	2601
Configure the popup list	2603
See Also	2605
Data binding in React Combo box component	2605
Binding local data.....	2605
Binding remote data	2611
See Also	2613
Value binding in ComboBox Component.....	2613
Primitive Data Types	2613
Object Data Types	2615
Templates in React Combo box component.....	2616
Item template	2616
Group template.....	2619
Header template	2622
Footer template	2625
No records template	2627
Action failure template	2629
See Also	2632
Virtualization in ComboBox Component	2632
Binding local data.....	2632
Binding remote data	2634
Grouping	2635

Filtering with Virtualization.....	2637
Grouping in React Combo box component.....	2638
Customization	2641
See Also	2641
Filtering in React Combo box component	2641
Limit the minimum filter character.....	2644
Change the filter type	2648
Case sensitive filtering	2652
Diacritics Filtering.....	2655
See Also	2657
Localization in React Combo box component	2657
Loading translations.....	2658
See Also	2661
Style in React Combo box component.....	2661
Customizing the appearance of wrapper element	2661
Customizing the dropdown icon's color	2661
Customizing the focus color.....	2662
Customizing the outline theme's focus color	2662
Customizing the disabled component's text color	2662
Customizing the float label element's focusing color	2662
Customizing the color of the placeholder text	2663
Customizing the text selection color.....	2663
Customizing the background color of focus, hover, and active item's	2663
Customizing the appearance of pop-up element	2663
Adding mandatory asterisk to placeholder and float label.....	2664
Accessibility in React Combo box component.....	2665
WAI-ARIA attributes.....	2666
Keyboard interaction	2667
Ensuring accessibility	2670
See also	2671
How To	2671
Autofill in React Combo box component.....	2671
Cascading in React Combo box component.....	2672
Icons support in React Combo box component.....	2675
Ej1 api migration in React Combo box component	2676

DataBinding.....	2676
Filtering	2677
Template	2678
Applying CSS.....	2678
Grouping	2679
Accessibility	2679
Placeholder	2679
Miscellaneous	2679
Sorting.....	2680
Selection.....	2680
Popup.....	2681
Common.....	2681
Context menu	2682
Getting Started.....	2682
Dependencies.....	2682
Setup your development environment	2683
Adding Syncfusion packages	2683
Adding Style sheet to the Application.....	2683
target {	2684
Add ContextMenu to the project.....	2684
Run the application	2686
See Also	2687
Icons and navigation in React Context menu component.....	2687
Icons	2687
Navigation URL.....	2688
See Also	2690
Template in React Context menu component.....	2690
Multilevel nesting	2692
See Also	2693
Accessibility in React ContextMenu component	2693
WAI-ARIA attributes.....	2694
Keyboard interaction	2695
Ensuring accessibility	2695
See also	2695
Style and appearance in React Context menu component	2695

How To	2696
Data binding in React Context menu component.....	2696
Render with separator in React Context menu component.....	2697
Open and close contextmenu in React Context menu component.....	2699
Change menu items dynamically in React Context menu component.....	2700
Scrollable contextmenu in React Context menu component.....	2702
Template in React Context menu component.....	2704
Underline a character in the item text in React Context menu component	2708
Open a dialog on contextmenu item click in React Context menu component	2710
Change animation settings in React Context menu component	2712
Add or remove context menu items in React Context menu component.....	2714
Enable or disable context menu items in React Context menu component	2717
Dashboard Layout	2719
Getting Started.....	2719
Dependencies.....	2719
Installation and configuration.....	2719
Adding Syncfusion packages	2720
Adding CSS Reference	2720
Add Dashboard Layout to the application	2720
Setting the `panels` property using HTML attributes	2720
Run the application	2724
Setting the `panels` property directly	2727
Setting size of cells in React Dashboard layout component	2731
Modifying cell size.....	2731
Setting cell spacing.....	2733
Graphical representation of layout.....	2734
Rendering component in right-to-left direction	2736
Panels.....	2737
Position sizing of panels in React Dashboard layout component.....	2737
Setting header of panels in React Dashboard layout component	2741
Add remove panels in React Dashboard layout component	2747
Interaction With Panels	2752
Dragging moving of panels in React Dashboard layout component.....	2752
Moving panels in React Dashboard layout component.....	2761
Resizing of panels in React Dashboard layout component.....	2763

Floating of panels in React Dashboard layout component	2768
Responsive adaptive in React Dashboard layout component	2770
State maintenance in React Dashboard layout component	2772
Style in React Dashboard layout component.....	2775
Customizing the dashboard layout panel header	2775
Customizing the dashboard layout panel content.....	2775
Customizing the dashboard layout panel resize icon	2775
Customizing the dashboard layout panel background	2775
Accessibility in React Dashboard Layout component	2776
WAI-ARIA attributes.....	2777
Keyboard interaction	2777
Ensuring accessibility	2777
See also	2777
DataManager	2777
Getting started	2777
Dependencies.....	2777
Installation and configuration.....	2778
Connection to a data source	2778
datatable {	2779
datatable td,.....	2779
datatable th {	2779
Filter	2793
Sort.....	2799
Page.....	2806
Component binding	2812
Data binding in React Data component	2819
Local data binding	2819
Remote data binding.....	2821
See Also	2823
Adaptors in React Data component.....	2823
Json adaptor	2823
Url adaptor	2825
OData adaptor	2826
ODataV4 adaptor	2827
Web API adaptor	2829

WebMethod Adaptor.....	2829
GraphQL Adaptor	2830
Writing custom adaptor.....	2835
Querying in React Data component.....	2838
Specifying resource name using `from`	2838
Projection using select	2840
Eager loading navigation properties	2842
Sorting.....	2844
Filtering	2847
Searching.....	2851
Grouping	2853
Paging.....	2857
Aggregation.....	2859
Hierarchical query.....	2862
Manipulation in React Data component.....	2866
Insert	2866
Update.....	2869
Remove	2872
Batch Edit Operation.....	2875
How to in React Data component.....	2879
Work in offline mode	2879
Sending additional parameters to server	2881
Adding custom headers	2883
DatePicker	2883
Getting started	2883
Dependencies.....	2884
Installation and configuration.....	2884
Adding Syncfusion packages	2884
Adding Style sheet to the Application.....	2885
Adding DatePicker component to the Application	2885
Run the application	2886
Setting the value, min and max dates.....	2887
See Also	2888
Date range in React Datepicker component.....	2889
Date format in React Datepicker component.....	2890

Date masking in React Datepicker component.....	2892
Configure Mask Placeholder	2895
Globalization in React Datepicker component	2897
Right-To-Left	3414
Strict mode in React Datepicker component.....	3740
Customization in React Datepicker component	3744
Adding mandatory asterisk to placeholder and float label.....	3746
See Also	3747
Date views in React Datepicker component.....	3748
Start view	3748
Depth view	3749
Accessibility in React Datepicker component.....	3750
WAI-ARIA attributes.....	3751
Keyboard Interaction	3752
Ensuring accessibility	3755
See also	3755
Style appearance in React Datepicker component.....	3755
Customizing the appearance of DatePicker wrapper element.....	3755
Customizing the DatePicker icon element.....	3755
Customizing the Calendar popup of the DatePicker.....	3756
Full screen mode support in mobiles and tablets.....	3756
How To	3758
Disabled the datepicker component in React Datepicker component.....	3758
Customize the datepicker day header in React Datepicker component	3758
Set the placeholder in React Datepicker component	3760
Set the readonly in React Datepicker component.....	3761
Open datepicker popup on input click in React Datepicker component.....	3762
Prevent the popup close in React Datepicker component	3763
Dynamic form validation in React Datepicker component.....	3763
Ej1 api migration in React Datepicker component	3769
Date Selection	3769
Date Format	3769
Calendar Views.....	3769
Date Range.....	3770
Disabled Dates	3770

Customization	3770
Accessibility	3772
Persistence	3773
Validation	3773
Common	3773
Globalization	3774
Strict Mode	3774
Open and Close	3774
View Navigation	3775
DateRangePicker	3775
Getting started	3775
Dependencies	3775
Installation and configuration	3776
Adding Syncfusion packages	3776
Adding Style sheet to the Application	3776
Adding DateRangePicker component to the Application	3777
Run the application	3778
Setting the start and end date in a range	3779
See Also	3780
Range selection in React Daterangepicker component	3780
Restrict the range within a range	3780
Range span	3782
Strict mode	3783
Globalization in React Daterangepicker component	3784
Right-To-Left	4302
Date Format	4629
Customization in React Daterangepicker component	4630
Day cell format	4630
First day of week	4632
Preset ranges	4633
See Also	4635
Accessibility in React Daterangepicker component	4635
WAI-ARIA attributes	4636
Keyboard Interaction	4636
Ensuring accessibility	4639

See also	4639
Style appearance in React Daterangepicker component	4639
Customizing the appearance of DateRangePicker wrapper element.....	4639
Customizing the DateRangePicker icon element.....	4640
Customizing the DateRangePicker popup calendar header	4640
Customizing the DateRangePicker popup calendar header title	4640
Customizing the DateRangePicker popup calendar content	4640
Customizing the DateRangePicker popup calendar content title.....	4641
Customizing the DateRangePicker popup calendar previous and next icon	4641
Customizing the DateRangePicker popup calendar date cell grid on hovering.....	4641
Customizing the DateRangePicker popup calendar primary button in footer	4641
Customizing the DateRangePicker popup calendar cancel button in footer.....	4642
Customizing the footer element in the DateRangePicker popup calendar	4642
Customizing the selected date cell grid in the DateRangePicker popup calendar	4642
Full screen mode support in mobiles and tablets.....	4643
How To	4645
Disable the daterangepicker component in React Daterangepicker component	4645
Customize the daterangepicker day header in React Daterangepicker component.....	4645
Set the placeholder in React Daterangepicker component.....	4647
Customization using cssclass in React Daterangepicker component	4648
Ej1 api migration in React Daterangepicker component	4649
Date Selection	4649
Date Format	4650
Date Range	4650
Disabled Dates	4650
Customization	4650
Accessibility.....	4651
Persistence	4651
Common.....	4652
Globalization	4653
Strict mode.....	4653
Open and Close	4653

Calendar

Getting started

This section explains you the steps required to create a simple Calendar and demonstrate the basic usage of the Calendar component.

To get start quickly with React Calendars, you can check on this video:

Dependencies

The below list of dependencies are required to use the `Calendar` component in your application.

```
`javascript
|-- @syncfusion/ej2-react-calendars
|-- @syncfusion/ej2-react-base
|-- @syncfusion/ej2-base
|-- @syncfusion/ej2-data
|-- @syncfusion/ej2-calendars
|-- @syncfusion/ej2-inputs
|-- @syncfusion/ej2-splitbuttons
|-- @syncfusion/ej2-lists
|-- @syncfusion/ej2-popups
|-- @syncfusion/ej2-buttons
`,`
```

Installation and configuration

You can use [create-react-app](#) to setup the applications. To install `create-react-app` run the following command.

```
`
npm install -g create-react-app
`,`
```

- To setup basic `React` sample use following commands.

```
`
create-react-app quickstart --scripts-version=react-scripts-ts
cd quickstart
`,`
```

Adding Syncfusion packages

All the available Essential JS 2 packages are published in [npmjs.com](#) public registry. You can choose the component that you want to install. For this application, we are going to use `Calendar` component.

To install Calendar component, use the following command


```
`bash
npm install @syncfusion/ej2-react-calendars --save
`
```

Adding Style sheet to the Application

To render the Calendar component, need to import Calendar and its dependent component's styles as given below in `App.css`.

```
`css
@import "../node_modules/@syncfusion/ej2-base/styles/material.css";
@import "../node_modules/@syncfusion/ej2-buttons/styles/material.css";
@import "../node_modules/@syncfusion/ej2-react-calendars/styles/material.css";
`
```

Note: If you want to refer the combined component styles, please make use of our [CRG](#) (Custom Resource Generator) in your application.

Adding Calendar component to the Application

- To include the Calendar component in application import the `CalendarComponent` from `ej2-react-calendars` package in `App.tsx`.
- Then add the Calendar component as shown in below code example.

[src/App.tsx]

[Class-component]

```
`ts
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import './App.css';
export default class App extends React.Component<{}, {}> {
  public render() {
    return <CalendarComponent id="calendar" />
  }
};
`
```

[Functional-component]

```
`ts
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
```

```
import * as ReactDOM from 'react-dom';

function App() {
  return <CalendarComponent id="calendar" />;
}

ReactDOM.render(<App />, document.getElementById('element'));
```

Run the application

Now run the `npm start` command in the console, it will run your application and open the browser window.

```
npm start
```

The below examples shows the basic calendar component.

[Class-component]

INDEX.JSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
  render() {
    return <CalendarComponent id="calendar"/>;
  }
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
  public render() {
    return <CalendarComponent id="calendar" />
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]

INDEX.JSX

```
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
```

```
import * as ReactDOM from 'react-dom';
function App() {
    return <CalendarComponent id="calendar"/>;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    return <CalendarComponent id="calendar" />;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

Setting the value, min and max dates

The following example demonstrates how to set the value, min and max dates on initializing the Calendar. Here the Calendar allows to select a date within a range from 9th to 15th in a month of May 2017. To know more about range restriction in Calendar, please refer this [page](#).

[Class-component]**INDEX.JSX**

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
    //initialize the value, min and max
    dateValue = new Date("05/11/2017");
    minDate = new Date("05/09/2017");
    maxDate = new Date("05/15/2017");
    render() {
        return <CalendarComponent id="calendar" value={this.dateValue}
min={this.minDate} max={this.maxDate}/>;
    }
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
    //initialize the value, min and max
    private dateValue: Date = new Date("05/11/2017");
    private minDate: Date = new Date("05/09/2017");
    private maxDate: Date = new Date("05/15/2017");
    public render() {
```

```

        return <CalendarComponent id="calendar" value={this.dateValue}
        min={this.minDate} max={this.maxDate} />
    };
    ReactDOM.render(<App />, document.getElementById('element'));

```

[Functional-component]

INDEX.JSX

```

import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    //initialize the value, min and max
    const dateValue = new Date("05/11/2017");
    const minDate = new Date("05/09/2017");
    const maxDate = new Date("05/15/2017");
    return <CalendarComponent id="calendar" value={dateValue} min={minDate}
    max={maxDate}/>;
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { CalendarComponent} from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    //initialize the value, min and max
    const dateValue: Date = new Date("05/11/2017");
    const minDate: Date = new Date("05/09/2017");
    const maxDate: Date = new Date("05/15/2017");
    return <CalendarComponent id="calendar" value={dateValue} min={minDate}
    max={maxDate} />
};
ReactDOM.render(<App />, document.getElementById('element'));

```

See Also

- [Select multiple dates in the Calendar](#)
- [Render Calendar with specific culture](#)
- [How to change the initial view of the Calendar](#)
- [Render Calendar with week numbers](#)
- [Show other month dates](#)

Date range in React Calendar component

Calendar provides an option to select a date value within a specified range by using the [min](#) and [max](#) properties. Always the min date has to be lesser than the max date.

The below example allows to select a date within a range from 7th to 27th dates in a month.

[Class-component]**INDEX.JSX**

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
    // creates a calendar with min and max property
    minDate = new Date(new Date().getFullYear(), new Date().getMonth(), 7);
    maxDate = new Date(new Date().getFullYear(), new Date().getMonth(), 27);
    dateValue = new Date(new Date().setDate(14));
    render() {
        return <CalendarComponent id="calendar" value={this.dateValue}
min={this.minDate} max={this.maxDate}/>;
    }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
    // creates a calendar with min and max property
    private minDate:Date= new Date(new Date().getFullYear(), new
Date().getMonth(), 7);
    private maxDate:Date= new Date(new Date().getFullYear(), new
Date().getMonth(), 27);
    private dateValue:Date= new Date(new Date().setDate(14));
    public render() {
        return <CalendarComponent id="calendar" value={this.dateValue}
min={this.minDate} max={this.maxDate} />
    }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]**INDEX.JSX**

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    // creates a calendar with min and max property
    const minDate = new Date(new Date().getFullYear(), new
Date().getMonth(), 7);
    const maxDate = new Date(new Date().getFullYear(), new
Date().getMonth(), 27);
    const dateValue = new Date(new Date().setDate(14));
```

```

    return <CalendarComponent id="calendar" value={dateValue} min={minDate}
    max={maxDate} />;
  }
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  // creates a calendar with min and max property
  const minDate:Date= new Date(new Date().getFullYear(), new
Date().getMonth(), 7);
  const maxDate:Date= new Date(new Date().getFullYear(), new
Date().getMonth(), 27);
  const dateValue:Date= new Date(new Date().setDate(14));
  return <CalendarComponent id="calendar" value={dateValue} min={minDate}
max={maxDate} />
};
ReactDOM.render(<App />, document.getElementById('element'));

```

If the value of `min` or `max` properties changed through code behind. Then you have to update the `value` property to set within the range. Or else, if the value is out of specified date range and less than `min` date, value property will be updated with min date or the value is higher than max date, value property will be updated with `max` date.

Globalization in React Calendar component

Globalization is the combination of internalization and localization. You can adapt the component to various languages by parsing and formatting the date or number ([Internationalization](#)), and also add culture specific customization and translation to the text ([localization](#)).

By default, Calendar date format, week and month names are specific to American English culture. It utilizes the [Essential JavaScript 2 Internationalization](#)

package to parse and format the date object based on the culture by using the official [UNICODE CLDR](#) JSON data and also it provides the [loadCldr](#) method to load the culture specific CLDR JSON data.

To go with the different culture other than `English`, follow the below steps.

- Install the `CLDR-Data` package by using the below command (it installs the CLDR JSON data). To know more about CLDR-Data refer the [CLDR-Data](#) link.

,

```
npm install cldr-data --save
```

,

Once the package installed, you can find the culture specific JSON data under the location `\node_modules\cldr-data`.

- Now import the installed CLDR JSON data into the `app.ts` file.
- Now use the [loadCldr](#) method to load the culture specific CLDR JSON data from the installed location to `app.ts` file.
- Calendar displayed `Sunday` as the first day of week based on default culture ("en-US"). If you want to display the Calendar with loaded culture's first day of week, you need to import `weekdata.json` file from the `cldr-data/supplemental` as given in the code example.

```
`ts
//import the loadCldr from ej2-base
import { loadCldr } from '@syncfusion/ej2-base';
import * as gregorian from 'cldr-data/main/de/ca-gregorian.json';
import * as numbers from 'cldr-data/main/de/numbers.json';
import * as timeZoneNames from 'cldr-data/main/de/timeZoneNames.json';
import * as numberingSystems from 'cldr-data/supplemental/numberingSystems.json';
import * as weekData from 'cldr-data/supplemental/weekData.json'; // To load the culture based first
day of week
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames, weekData);
`
```

if you are facing the error `/node_modules/cldr-data/main/de/*.json (1,1): unused expression, expected an assignment or function call` when you are adding the json files to render the culture sample, then add the below configuration in your `tslint.json` file

```
`ts
"linterOptions": {
  "exclude": [
    "*.json",
    "/*.json"
  ]
}
`
```

The `Localization` library allows you to localize default text content of the Calendar. The Calendar component has static text for **today** feature that can be changed to other cultures (Arabic, Deutsch, French, etc.) by defining the

[locale](#) value and translation object.

Locale keywords | Text

today | Name of the button to choose Today date.

- Before changing to a culture other than English, ensure that locale text for the concerned culture is loaded through `load` method of `L10n` class.

```
`ts
//Load the L10n, loadCldr from ej2-base
import { L10n } from "@syncfusion/ej2-base";
//load the locale object to set the localized placeholder value
L10n.load({
  'de': {
    'calendar': { today:'heute' }
  }
});
`
```

- Set the culture by using the [locale](#) property.

The following example demonstrates the Calendar in **German** culture.

[Class-component]

CA-GREGORIAN.JSON

```
{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 12879 $",
          "_cldrVersion": "30.0.3"
        },
        "language": "de"
      },
      "dates": {
        "calendars": {
          "gregorian": {
            "months": {
              "format": {
                "abbreviated": {
                  "1": "Jan.",
                  "2": "Feb.",
                  "3": "März",
                  "4": "Apr.",
                  "5": "Mai",
                  "6": "Juni",
                  "7": "Juli",
                  "8": "Aug.",
                  "9": "Sep.",
                  "10": "Okt.",
                  "11": "Nov.",

```



```

        "12": "Dez."
    },
    "narrow": {
        "1": "J",
        "2": "F",
        "3": "M",
        "4": "A",
        "5": "M",
        "6": "J",
        "7": "J",
        "8": "A",
        "9": "S",
        "10": "O",
        "11": "N",
        "12": "D"
    },
    "wide": {
        "1": "Januar",
        "2": "Februar",
        "3": "März",
        "4": "April",
        "5": "Mai",
        "6": "Juni",
        "7": "Juli",
        "8": "August",
        "9": "September",
        "10": "Oktober",
        "11": "November",
        "12": "Dezember"
    }
},
"stand-alone": {
    "abbreviated": {
        "1": "Jan",
        "2": "Feb",
        "3": "Mär",
        "4": "Apr",
        "5": "Mai",
        "6": "Jun",
        "7": "Jul",
        "8": "Aug",
        "9": "Sep",
        "10": "Okt",
        "11": "Nov",
        "12": "Dez"
    },
    "narrow": {
        "1": "J",
        "2": "F",
        "3": "M",
        "4": "A",
        "5": "M",
        "6": "J",
        "7": "J",
        "8": "A",
        "9": "S",
        "10": "O",

```

```

        "11": "N",
        "12": "D"
    },
    "wide": {
        "1": "Januar",
        "2": "Februar",
        "3": "März",
        "4": "April",
        "5": "Mai",
        "6": "Juni",
        "7": "Juli",
        "8": "August",
        "9": "September",
        "10": "Oktober",
        "11": "November",
        "12": "Dezember"
    }
    },
    "days": {
        "format": {
            "abbreviated": {
                "sun": "So.",
                "mon": "Mo.",
                "tue": "Di.",
                "wed": "Mi.",
                "thu": "Do.",
                "fri": "Fr.",
                "sat": "Sa."
            },
            "narrow": {
                "sun": "S",
                "mon": "M",
                "tue": "D",
                "wed": "M",
                "thu": "D",
                "fri": "F",
                "sat": "S"
            },
            "short": {
                "sun": "So.",
                "mon": "Mo.",
                "tue": "Di.",
                "wed": "Mi.",
                "thu": "Do.",
                "fri": "Fr.",
                "sat": "Sa."
            },
            "wide": {
                "sun": "Sonntag",
                "mon": "Montag",
                "tue": "Dienstag",
                "wed": "Mittwoch",
                "thu": "Donnerstag",
                "fri": "Freitag",
                "sat": "Samstag"
            }
        }
    }
}

```

```

    },
    "stand-alone": {
      "abbreviated": {
        "sun": "So",
        "mon": "Mo",
        "tue": "Di",
        "wed": "Mi",
        "thu": "Do",
        "fri": "Fr",
        "sat": "Sa"
      },
      "narrow": {
        "sun": "S",
        "mon": "M",
        "tue": "D",
        "wed": "M",
        "thu": "D",
        "fri": "F",
        "sat": "S"
      },
      "short": {
        "sun": "So.",
        "mon": "Mo.",
        "tue": "Di.",
        "wed": "Mi.",
        "thu": "Do.",
        "fri": "Fr.",
        "sat": "Sa."
      },
      "wide": {
        "sun": "Sonntag",
        "mon": "Montag",
        "tue": "Dienstag",
        "wed": "Mittwoch",
        "thu": "Donnerstag",
        "fri": "Freitag",
        "sat": "Samstag"
      }
    }
  },
  "quarters": {
    "format": {
      "abbreviated": {
        "1": "Q1",
        "2": "Q2",
        "3": "Q3",
        "4": "Q4"
      },
      "narrow": {
        "1": "1",
        "2": "2",
        "3": "3",
        "4": "4"
      },
      "wide": {
        "1": "1. Quartal",
        "2": "2. Quartal",

```

```

        "3": "3. Quartal",
        "4": "4. Quartal"
    },
    },
    "stand-alone": {
        "abbreviated": {
            "1": "Q1",
            "2": "Q2",
            "3": "Q3",
            "4": "Q4"
        },
        "narrow": {
            "1": "1",
            "2": "2",
            "3": "3",
            "4": "4"
        },
        "wide": {
            "1": "1. Quartal",
            "2": "2. Quartal",
            "3": "3. Quartal",
            "4": "4. Quartal"
        }
    },
    },
    "dayPeriods": {
        "format": {
            "abbreviated": {
                "midnight": "Mitternacht",
                "am": "vorm.",
                "pm": "nachm.",
                "morning1": "morgens",
                "morning2": "vormittags",
                "afternoon1": "mittags",
                "afternoon2": "nachmittags",
                "evening1": "abends",
                "night1": "nachts"
            },
            "narrow": {
                "midnight": "Mitternacht",
                "am": "vm.",
                "pm": "nm.",
                "morning1": "morgens",
                "morning2": "vormittags",
                "afternoon1": "mittags",
                "afternoon2": "nachmittags",
                "evening1": "abends",
                "night1": "nachts"
            },
            "wide": {
                "midnight": "Mitternacht",
                "am": "vorm.",
                "pm": "nachm.",
                "morning1": "morgens",
                "morning2": "vormittags",
                "afternoon1": "mittags",
                "afternoon2": "nachmittags",
            }
        }
    }
}

```

```

        "evening1": "abends",
        "night1": "nachts"
    },
    },
    "stand-alone": {
        "abbreviated": {
            "midnight": "Mitternacht",
            "am": "vorm.",
            "pm": "nachm.",
            "morning1": "Morgen",
            "morning2": "Vormittag",
            "afternoon1": "Mittag",
            "afternoon2": "Nachmittag",
            "evening1": "Abend",
            "night1": "Nacht"
        },
        "narrow": {
            "midnight": "Mitternacht",
            "am": "vorm.",
            "pm": "nachm.",
            "morning1": "Morgen",
            "morning2": "Vormittag",
            "afternoon1": "Mittag",
            "afternoon2": "Nachmittag",
            "evening1": "Abend",
            "night1": "Nacht"
        },
        "wide": {
            "midnight": "Mitternacht",
            "am": "vorm.",
            "pm": "nachm.",
            "morning1": "Morgen",
            "morning2": "Vormittag",
            "afternoon1": "Mittag",
            "afternoon2": "Nachmittag",
            "evening1": "Abend",
            "night1": "Nacht"
        }
    },
    },
    "eras": {
        "eraNames": {
            "0": "v. Chr.",
            "0-alt-variant": "vor unserer Zeitrechnung",
            "1": "n. Chr.",
            "1-alt-variant": "unserer Zeitrechnung"
        },
        "eraAbbr": {
            "0": "v. Chr.",
            "0-alt-variant": "v. u. Z.",
            "1": "n. Chr.",
            "1-alt-variant": "u. Z."
        },
        "eraNarrow": {
            "0": "v. Chr.",
            "0-alt-variant": "v. u. Z.",
            "1": "n. Chr.",

```

```

        "1-alt-variant": "u. Z."
    },
    },
    "dateFormats": {
        "full": "EEEE, d. MMMM y",
        "long": "d. MMMM y",
        "medium": "dd.MM.y",
        "short": "dd.MM.yy"
    },
    "timeFormats": {
        "full": "HH:mm:ss zzzz",
        "long": "HH:mm:ss z",
        "medium": "HH:mm:ss",
        "short": "HH:mm"
    },
    "dateTimeFormats": {
        "full": "{1} 'um' {0}",
        "long": "{1} 'um' {0}",
        "medium": "{1}, {0}",
        "short": "{1}, {0}",
        "availableFormats": {
            "d": "d",
            "E": "ccc",
            "Ed": "E, d.",
            "Ehm": "E h:mm a",
            "EHm": "E, HH:mm",
            "Ehms": "E, h:mm:ss a",
            "EHms": "E, HH:mm:ss",
            "Gy": "y G",
            "GyMMM": "MMM y G",
            "GyMMMd": "d. MMM y G",
            "GyMMMED": "E, d. MMM y G",
            "h": "h 'Uhr' a",
            "H": "HH 'Uhr'",
            "hm": "h:mm a",
            "Hm": "HH:mm",
            "hms": "h:mm:ss a",
            "Hms": "HH:mm:ss",
            "hmsv": "h:mm:ss a v",
            "Hmsv": "HH:mm:ss v",
            "hmv": "h:mm a v",
            "Hmv": "HH:mm v",
            "M": "L",
            "Md": "d.M.",
            "MEd": "E, d.M.",
            "MMd": "d.MM.",
            "MMdd": "dd.MM.",
            "MMM": "LLL",
            "MMMd": "d. MMM",
            "MMMED": "E, d. MMM",
            "MMMMd": "d. MMMM",
            "MMMMEd": "E, d. MMMM",
            "MMMMW": "'Woche' W 'im' MMM",
            "MMMMW": "'Woche' W 'im' MMM",
            "ms": "mm:ss",
            "Y": "Y",
            "yM": "M.y",

```

```

        "yMd": "d.M.y",
        "yMEd": "E, d.M.y",
        "yMM": "MM.y",
        "yMMdd": "dd.MM.y",
        "yMMM": "MMM y",
        "yMMMd": "d. MMM y",
        "yMMMEd": "E, d. MMM y",
        "yMMMM": "MMMM y",
        "yQQQ": "QQQ y",
        "yQQQQ": "QQQQ y",
        "yw": "'Woche' w 'des' 'Jahres' y",
        "yw": "'Woche' w 'des' 'Jahres' y"
    },
    "appendItems": {
        "Day": "{0} ({2}: {1})",
        "Day-Of-Week": "{0} {1}",
        "Era": "{1} {0}",
        "Hour": "{0} ({2}: {1})",
        "Minute": "{0} ({2}: {1})",
        "Month": "{0} ({2}: {1})",
        "Quarter": "{0} ({2}: {1})",
        "Second": "{0} ({2}: {1})",
        "Timezone": "{0} {1}",
        "Week": "{0} ({2}: {1})",
        "Year": "{1} {0}"
    },
    "intervalFormats": {
        "intervalFormatFallback": "{0} - {1}",
        "d": {
            "d": "d.-d."
        },
        "h": {
            "a": "h 'Uhr' a - h 'Uhr' a",
            "h": "h - h 'Uhr' a"
        },
        "H": {
            "H": "HH-HH 'Uhr'"
        },
        "hm": {
            "a": "h:mm a - h:mm a",
            "h": "h:mm-h:mm a",
            "m": "h:mm-h:mm a"
        },
        "Hm": {
            "H": "HH:mm-HH:mm 'Uhr'",
            "m": "HH:mm-HH:mm 'Uhr'"
        },
        "hmv": {
            "a": "h:mm a - h:mm a v",
            "h": "h:mm-h:mm a v",
            "m": "h:mm-h:mm a v"
        },
        "Hmv": {
            "H": "HH:mm-HH:mm 'Uhr' v",
            "m": "HH:mm-HH:mm 'Uhr' v"
        },
        "hv": {

```

```

    "a": "h a - h a v",
    "h": "h-h a v"
  },
  "Hv": {
    "H": "HH-HH 'Uhr' v"
  },
  "M": {
    "M": "M.-M."
  },
  "Md": {
    "d": "dd.MM. - dd.MM.",
    "M": "dd.MM. - dd.MM."
  },
  "MEd": {
    "d": "E, dd.MM. - E, dd.MM.",
    "M": "E, dd.MM. - E, dd.MM."
  },
  "MMM": {
    "M": "MMM-MMM"
  },
  "MMMd": {
    "d": "d.-d. MMM",
    "M": "d. MMM - d. MMM"
  },
  "MMMEd": {
    "d": "E, d. - E, d. MMM",
    "M": "E, d. MMM - E, d. MMM"
  },
  "MMMM": {
    "M": "LLLL-LLLL"
  },
  "Y": {
    "Y": "Y-Y"
  },
  "YM": {
    "M": "MM.y - MM.y",
    "Y": "MM.y - MM.y"
  },
  "YMd": {
    "d": "dd.MM.y - dd.MM.y",
    "M": "dd.MM.y - dd.MM.y",
    "Y": "dd.MM.y - dd.MM.y"
  },
  "YMEd": {
    "d": "E, dd.MM.y - E, dd.MM.y",
    "M": "E, dd.MM.y - E, dd.MM.y",
    "Y": "E, dd.MM.y - E, dd.MM.y"
  },
  "YMMM": {
    "M": "MMM-MMM y",
    "Y": "MMM y - MMM y"
  },
  "YMMMd": {
    "d": "d.-d. MMM y",
    "M": "d. MMM - d. MMM y",
    "Y": "d. MMM y - d. MMM y"
  },
  },

```



```
"yMMMMEd": {  
    "d": "E, d. - E, d. MMM y",  
    "M": "E, d. MMM - E, d. MMM y",  
    "y": "E, d. MMM y - E, d. MMM y"  
},  
"yMMMM": {  
    "M": "MMMM-MMMM y",  
    "y": "MMMM y - MMMM y"  
}  
  
}  
  
}  
  
}  
  
}  
  
}  
  
}
```

CA-GREGORIAN.JSX

```
{
  "main";
  {
    "de";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 12879 $",
          "_cldrVersion";
          "30.0.3";
        }
        "language";
        "de";
      }
    }
    "dates";
    {
      "calendars";
      {
        "gregorian";
        {
          "months";
          {
            "format";
            {
              "abbreviated";
              {
                "1";
                "Jan.",
                "2";
                "Feb.",
                "3";
                "März",
                "4";
              }
            }
          }
        }
      }
    }
  }
}
```

```

        "Apr.",
        "5";
        "Mai",
        "6";
        "Juni",
        "7";
        "Juli",
        "8";
        "Aug.",
        "9";
        "Sep.",
        "10";
        "Okt.",
        "11";
        "Nov.",
        "12";
        "Dez.";
    }
    "narrow";
    {
        "1";
        "J",
        "2";
        "F",
        "3";
        "M",
        "4";
        "A",
        "5";
        "M",
        "6";
        "J",
        "7";
        "J",
        "8";
        "A",
        "9";
        "S",
        "10";
        "O",
        "11";
        "N",
        "12";
        "D";
    }
    "wide";
    {
        "1";
        "Januar",
        "2";
        "Februar",
        "3";
        "März",
        "4";
        "April",
        "5";
        "Mai",

```

```
        "6";
        "Juni",
        "7";
        "Juli",
        "8";
        "August",
        "9";
        "September",
        "10";
        "Oktober",
        "11";
        "November",
        "12";
        "Dezember";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
        "Jan",
        "2";
        "Feb",
        "3";
        "Mär",
        "4";
        "Apr",
        "5";
        "Mai",
        "6";
        "Jun",
        "7";
        "Jul",
        "8";
        "Aug",
        "9";
        "Sep",
        "10";
        "Okt",
        "11";
        "Nov",
        "12";
        "Dez";
    }
}
"narrow";
{
    "1";
    "J",
    "2";
    "F",
    "3";
    "M",
    "4";
    "A",
    "5";
    "M",
```

```

        "6";
        "J",
        "7";
        "J",
        "8";
        "A",
        "9";
        "S",
        "10";
        "O",
        "11";
        "N",
        "12";
        "D";
    }
    "wide";
    {
        "1";
        "Januar",
        "2";
        "Februar",
        "3";
        "März",
        "4";
        "April",
        "5";
        "Mai",
        "6";
        "Juni",
        "7";
        "Juli",
        "8";
        "August",
        "9";
        "September",
        "10";
        "Oktober",
        "11";
        "November",
        "12";
        "Dezember";
    }
}
"days";
{
    "format";
    {
        "abbreviated";
        {
            "sun";
            "So.",
            "mon";
            "Mo.",
            "tue";
            "Di.",
            "wed";

```

```
        "Mi.",
        "thu";
        "Do.",
        "fri";
        "Fr.",
        "sat";
        "Sa.";
    }
    "narrow";
    {
        "sun";
        "S",
        "mon";
        "M",
        "tue";
        "D",
        "wed";
        "M",
        "thu";
        "D",
        "fri";
        "F",
        "sat";
        "S";
    }
    "short";
    {
        "sun";
        "So.",
        "mon";
        "Mo.",
        "tue";
        "Di.",
        "wed";
        "Mi.",
        "thu";
        "Do.",
        "fri";
        "Fr.",
        "sat";
        "Sa.";
    }
    "wide";
    {
        "sun";
        "Sonntag",
        "mon";
        "Montag",
        "tue";
        "Dienstag",
        "wed";
        "Mittwoch",
        "thu";
        "Donnerstag",
        "fri";
        "Freitag",
        "sat";
    }
```

```

        "Samstag";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "sun";
        "So",
        "mon";
        "Mo",
        "tue";
        "Di",
        "wed";
        "Mi",
        "thu";
        "Do",
        "fri";
        "Fr",
        "sat";
        "Sa";
    }
    "narrow";
    {
        "sun";
        "S",
        "mon";
        "M",
        "tue";
        "D",
        "wed";
        "M",
        "thu";
        "D",
        "fri";
        "F",
        "sat";
        "S";
    }
    "short";
    {
        "sun";
        "So.",
        "mon";
        "Mo.",
        "tue";
        "Di.",
        "wed";
        "Mi.",
        "thu";
        "Do.",
        "fri";
        "Fr.",
        "sat";
        "Sa.";
    }
    "wide";

```

```

        {
            "sun";
            "Sonntag",
            "mon";
            "Montag",
            "tue";
            "Dienstag",
            "wed";
            "Mittwoch",
            "thu";
            "Donnerstag",
            "fri";
            "Freitag",
            "sat";
            "Samstag";
        }
    }
}
"quarters";
{
    "format";
    {
        "abbreviated";
        {
            "1";
            "Q1",
            "2";
            "Q2",
            "3";
            "Q3",
            "4";
            "Q4";
        }
        "narrow";
        {
            "1";
            "1",
            "2";
            "2",
            "3";
            "3",
            "4";
            "4";
        }
        "wide";
        {
            "1";
            "1. Quartal",
            "2";
            "2. Quartal",
            "3";
            "3. Quartal",
            "4";
            "4. Quartal";
        }
    }
}
"stand-alone";

```

```

        {
            "abbreviated";
            {
                "1";
                "Q1",
                "2";
                "Q2",
                "3";
                "Q3",
                "4";
                "Q4";
            }
            "narrow";
            {
                "1";
                "1",
                "2";
                "2",
                "3";
                "3",
                "4";
                "4";
            }
            "wide";
            {
                "1";
                "1. Quartal",
                "2";
                "2. Quartal",
                "3";
                "3. Quartal",
                "4";
                "4. Quartal";
            }
        }
    }
    "dayPeriods";
    {
        "format";
        {
            "abbreviated";
            {
                "midnight";
                "Mitternacht",
                "am";
                "vorm.",
                "pm";
                "nachm.",
                "morning1";
                "morgens",
                "morning2";
                "vormittags",
                "afternoon1";
                "mittags",
                "afternoon2";
                "nachmittags",
                "evening1";
            }
        }
    }

```



```

        "abends",
        "night1";
        "nachts";
    }
    "narrow";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vm.",
        "pm";
        "nm.",
        "morning1";
        "morgens",
        "morning2";
        "vormittags",
        "afternoon1";
        "mittags",
        "afternoon2";
        "nachmittags",
        "evening1";
        "abends",
        "night1";
        "nachts";
    }
    "wide";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vorm.",
        "pm";
        "nachm.",
        "morning1";
        "morgens",
        "morning2";
        "vormittags",
        "afternoon1";
        "mittags",
        "afternoon2";
        "nachmittags",
        "evening1";
        "abends",
        "night1";
        "nachts";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vorm.",
        "pm";
        "nachm.",

```

```

        "morning1";
        "Morgen",
        "morning2";
        "Vormittag",
        "afternoon1";
        "Mittag",
        "afternoon2";
        "Nachmittag",
        "evening1";
        "Abend",
        "night1";
        "Nacht";
    }
    "narrow";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vorm.",
        "pm";
        "nachm.",
        "morning1";
        "Morgen",
        "morning2";
        "Vormittag",
        "afternoon1";
        "Mittag",
        "afternoon2";
        "Nachmittag",
        "evening1";
        "Abend",
        "night1";
        "Nacht";
    }
    "wide";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vorm.",
        "pm";
        "nachm.",
        "morning1";
        "Morgen",
        "morning2";
        "Vormittag",
        "afternoon1";
        "Mittag",
        "afternoon2";
        "Nachmittag",
        "evening1";
        "Abend",
        "night1";
        "Nacht";
    }
}

```

```

    "eras";
    {
        "eraNames";
        {
            "0";
            "v. Chr.",
            "0-alt-variant";
            "vor unserer Zeitrechnung",
            "1";
            "n. Chr.",
            "1-alt-variant";
            "unserer Zeitrechnung";
        }
        "eraAbbr";
        {
            "0";
            "v. Chr.",
            "0-alt-variant";
            "v. u. Z.",
            "1";
            "n. Chr.",
            "1-alt-variant";
            "u. Z.";
        }
        "eraNarrow";
        {
            "0";
            "v. Chr.",
            "0-alt-variant";
            "v. u. Z.",
            "1";
            "n. Chr.",
            "1-alt-variant";
            "u. Z.";
        }
    }
    "dateFormats";
    {
        "full";
        "EEEE, d. MMMM y",
        "long";
        "d. MMMM y",
        "medium";
        "dd.MM.y",
        "short";
        "dd.MM.yy";
    }
    "timeFormats";
    {
        "full";
        "HH:mm:ss zzzz",
        "long";
        "HH:mm:ss z",
        "medium";
        "HH:mm:ss",
        "short";
        "HH:mm";
    }

```

```

    }
    "dateTimeFormats";
    {
        "full";
        "{1} 'um' {0}",
        "long";
        "{1} 'um' {0}",
        "medium";
        "{1}, {0}",
        "short";
        "{1}, {0}",
        "availableFormats";
        {
            "d";
            "d",
            "E";
            "ccc",
            "Ed";
            "E, d.",
            "Ehm";
            "E h:mm a",
            "EHm";
            "E, HH:mm",
            "Ehms";
            "E, h:mm:ss a",
            "EHms";
            "E, HH:mm:ss",
            "Gy";
            "y G",
            "GyMMM";
            "MMM y G",
            "GyMMMd";
            "d. MMM y G",
            "GyMMMED";
            "E, d. MMM y G",
            "h";
            "h 'Uhr' a",
            "H";
            "HH 'Uhr'",
            "hm";
            "h:mm a",
            "Hm";
            "HH:mm",
            "hms";
            "h:mm:ss a",
            "Hms";
            "HH:mm:ss",
            "hmsv";
            "h:mm:ss a v",
            "Hmsv";
            "HH:mm:ss v",
            "hmv";
            "h:mm a v",
            "Hmv";
            "HH:mm v",
            "M";
            "L",

```

```

        "Md";
        "d.M.",
        "MEd";
        "E, d.M.",
        "MMd";
        "d.MM.",
        "MMdd";
        "dd.MM.",
        "MMM";
        "LLL",
        "MMMd";
        "d. MMM",
        "MMMEd";
        "E, d. MMM",
        "MMMMd";
        "d. MMMM",
        "MMMMEd";
        "E, d. MMMM",
        "MMMMW";
        "'Woche' W 'im' MMM",
        "MMMMW";
        "'Woche' W 'im' MMM",
        "ms";
        "mm:ss",
        "y";
        "Y",
        "yM";
        "M.y",
        "yMd";
        "d.M.y",
        "yMEd";
        "E, d.M.y",
        "yMM";
        "MM.y",
        "yMMdd";
        "dd.MM.y",
        "yMMM";
        "MMM y",
        "yMMMd";
        "d. MMM y",
        "yMMMEd";
        "E, d. MMM y",
        "yMMMM";
        "MMMM y",
        "yQQQ";
        "QQQ y",
        "yQQQQ";
        "QQQQ y",
        "yw";
        "'Woche' w 'des' 'Jahres' y",
        "yw";
        "'Woche' w 'des' 'Jahres' y";
    }
    "appendItems";
    {
        "Day";
        "{0} ({2}: {1})",

```

```

        "Day-Of-Week";
        "{0} {1}",
        "Era";
        "{1} {0}",
        "Hour";
        "{0} ({2}: {1})",
        "Minute";
        "{0} ({2}: {1})",
        "Month";
        "{0} ({2}: {1})",
        "Quarter";
        "{0} ({2}: {1})",
        "Second";
        "{0} ({2}: {1})",
        "Timezone";
        "{0} {1}",
        "Week";
        "{0} ({2}: {1})",
        "Year";
        "{1} {0}";
    }
    "intervalFormats";
    {
        "intervalFormatFallback";
        "{0} - {1}",
        "d";
        {
            "d";
            "d.-d.";
        }
        "h";
        {
            "a";
            "h 'Uhr' a - h 'Uhr' a",
            "h";
            "h - h 'Uhr' a";
        }
        "H";
        {
            "H";
            "HH-HH 'Uhr'";
        }
        "hm";
        {
            "a";
            "h:mm a - h:mm a",
            "h";
            "h:mm-h:mm a",
            "m";
            "h:mm-h:mm a";
        }
        "Hm";
        {
            "H";
            "HH:mm-HH:mm 'Uhr'",
            "m";
            "HH:mm-HH:mm 'Uhr'";
        }
    }

```

```

    }
    "hmv";
    {
        "a";
        "h:mm a - h:mm a v",
        "h";
        "h:mm-h:mm a v",
        "m";
        "h:mm-h:mm a v";
    }
    "Hmv";
    {
        "H";
        "HH:mm-HH:mm 'Uhr' v",
        "m";
        "HH:mm-HH:mm 'Uhr' v";
    }
    "hv";
    {
        "a";
        "h a - h a v",
        "h";
        "h-h a v";
    }
    "Hv";
    {
        "H";
        "HH-HH 'Uhr' v";
    }
    "M";
    {
        "M";
        "M.-M.";
    }
    "Md";
    {
        "d";
        "dd.MM. - dd.MM.",
        "M";
        "dd.MM. - dd.MM.";
    }
    "MEd";
    {
        "d";
        "E, dd.MM. - E, dd.MM.",
        "M";
        "E, dd.MM. - E, dd.MM.";
    }
    "MMM";
    {
        "M";
        "MMM-MMM";
    }
    "MMMd";
    {
        "d";
        "d.-d. MMM",

```

```

        "M";
        "d. MMM - d. MMM";
    }
    "MMEd";
    {
        "d";
        "E, d. - E, d. MMM",
        "M";
        "E, d. MMM - E, d. MMM";
    }
    "MMMM";
    {
        "M";
        "LLLL-LLLL";
    }
    "Y";
    {
        "Y";
        "Y-Y";
    }
    "YM";
    {
        "M";
        "MM.Y - MM.Y",
        "Y";
        "MM.Y - MM.Y";
    }
    "YMd";
    {
        "d";
        "dd.MM.Y - dd.MM.Y",
        "M";
        "dd.MM.Y - dd.MM.Y",
        "Y";
        "dd.MM.Y - dd.MM.Y";
    }
    "YMEd";
    {
        "d";
        "E, dd.MM.Y - E, dd.MM.Y",
        "M";
        "E, dd.MM.Y - E, dd.MM.Y",
        "Y";
        "E, dd.MM.Y - E, dd.MM.Y";
    }
    "YMMM";
    {
        "M";
        "MMM-MMM Y",
        "Y";
        "MMM Y - MMM Y";
    }
    "YMMMd";
    {
        "d";
        "d.-d. MMM Y",
        "M";
    }

```



```

        "d. MMM - d. MMM y",
        "y";
    "d. MMM y - d. MMM y";
}
"yMMMEd";
{
    "d";
    "E, d. - E, d. MMM y",
    "M";
    "E, d. MMM - E, d. MMM y",
    "y";
    "E, d. MMM y - E, d. MMM y";
}
"yMMMM";
{
    "M";
    "MMMM-MMMM y",
    "y";
    "MMMM y - MMMM y";
}
}
}
}
}
}
}
```

CURRENCIES.JSON

```
{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31"
        },
        "language": "de"
      },
      "numbers": {
        "currencies": {
          "ADP": {
            "displayName": "Andorranische Pesete",
            "displayName-count-one": "Andorranische Pesete",
            "displayName-count-other": "Andorranische Peseten",
            "symbol": "ADP"
          },
          "AED": {
            "displayName": "VAE-Dirham",
            "displayName-count-one": "VAE-Dirham",
            "displayName-count-other": "VAE-Dirham",
            "symbol": "AED"
          },
          "AFA": {
```

```

        "displayName": "Afghanische Afghani (1927-2002)",
        "displayName-count-one": "Afghanische Afghani (1927-2002)",
        "displayName-count-other": "Afghanische Afghani (1927-2002)",
        "symbol": "AFA"
    },
    "AFN": {
        "displayName": "Afghanischer Afghani",
        "displayName-count-one": "Afghanischer Afghani",
        "displayName-count-other": "Afghanische Afghani",
        "symbol": "AFN"
    },
    "ALK": {
        "displayName": "Albanischer Lek (1946-1965)",
        "displayName-count-one": "Albanischer Lek (1946-1965)",
        "displayName-count-other": "Albanische Lek (1946-1965)"
    },
    "ALL": {
        "displayName": "Albanischer Lek",
        "displayName-count-one": "Albanischer Lek",
        "displayName-count-other": "Albanische Lek",
        "symbol": "ALL"
    },
    "AMD": {
        "displayName": "Armenischer Dram",
        "displayName-count-one": "Armenischer Dram",
        "displayName-count-other": "Armenische Dram",
        "symbol": "AMD"
    },
    "ANG": {
        "displayName": "Niederländische-Antillen-Gulden",
        "displayName-count-one": "Niederländische-Antillen-Gulden",
        "displayName-count-other": "Niederländische-Antillen-Gulden",
        "symbol": "ANG"
    },
    "AOA": {
        "displayName": "Angolanischer Kwanza",
        "displayName-count-one": "Angolanischer Kwanza",
        "displayName-count-other": "Angolanische Kwanza",
        "symbol": "AOA",
        "symbol-alt-narrow": "Kz"
    },
    "AOK": {
        "displayName": "Angolanischer Kwanza (1977-1990)",
        "displayName-count-one": "Angolanischer Kwanza (1977-1990)",
        "displayName-count-other": "Angolanische Kwanza (1977-1990)",
        "symbol": "AOK"
    },
    "AON": {
        "displayName": "Angolanischer Neuer Kwanza (1990-2000)",
        "displayName-count-one": "Angolanischer Neuer Kwanza (1990-2000)",
        "displayName-count-other": "Angolanische Neue Kwanza (1990-2000)",
        "symbol": "AON"
    },
    "AOR": {
        "displayName": "Angolanischer Kwanza Reajustado (1995-1999)",

```

```

        "displayName-count-one": "Angolanischer Kwanza Reajustado (1995-1999)",
        "displayName-count-other": "Angolanische Kwanza Reajustado (1995-1999)",
        "symbol": "AOR"
    },
    "ARA": {
        "displayName": "Argentinischer Austral",
        "displayName-count-one": "Argentinischer Austral",
        "displayName-count-other": "Argentinische Austral",
        "symbol": "ARA"
    },
    "ARL": {
        "displayName": "Argentinischer Peso Ley (1970-1983)",
        "displayName-count-one": "Argentinischer Peso Ley (1970-1983)",
        "displayName-count-other": "Argentinische Pesos Ley (1970-1983)",
        "symbol": "ARL"
    },
    "ARM": {
        "displayName": "Argentinischer Peso (1881-1970)",
        "displayName-count-one": "Argentinischer Peso (1881-1970)",
        "displayName-count-other": "Argentinische Pesos (1881-1970)",
        "symbol": "ARM"
    },
    "ARP": {
        "displayName": "Argentinischer Peso (1983-1985)",
        "displayName-count-one": "Argentinischer Peso (1983-1985)",
        "displayName-count-other": "Argentinische Peso (1983-1985)",
        "symbol": "ARP"
    },
    "ARS": {
        "displayName": "Argentinischer Peso",
        "displayName-count-one": "Argentinischer Peso",
        "displayName-count-other": "Argentinische Pesos",
        "symbol": "ARS",
        "symbol-alt-narrow": "$"
    },
    "ATS": {
        "displayName": "Österreichischer Schilling",
        "displayName-count-one": "Österreichischer Schilling",
        "displayName-count-other": "Österreichische Schilling",
        "symbol": "öS"
    },
    "AUD": {
        "displayName": "Australischer Dollar",
        "displayName-count-one": "Australischer Dollar",
        "displayName-count-other": "Australische Dollar",
        "symbol": "AU$",
        "symbol-alt-narrow": "$"
    },
    "AWG": {
        "displayName": "Aruba-Florin",
        "displayName-count-one": "Aruba-Florin",
        "displayName-count-other": "Aruba-Florin",
        "symbol": "AWG"
    },
    },

```

```

    "AZM": {
      "displayName": "Aserbajdschan-Manat (1993-2006)",
      "displayName-count-one": "Aserbajdschan-Manat (1993-2006)",
      "displayName-count-other": "Aserbajdschan-Manat (1993-2006)",
      "symbol": "AZM"
    },
    "AZN": {
      "displayName": "Aserbajdschan-Manat",
      "displayName-count-one": "Aserbajdschan-Manat",
      "displayName-count-other": "Aserbajdschan-Manat",
      "symbol": "AZN"
    },
    "BAD": {
      "displayName": "Bosnien und Herzegowina Dinar (1992-1994)",
      "displayName-count-one": "Bosnien und Herzegowina Dinar (1992-1994)",
      "displayName-count-other": "Bosnien und Herzegowina Dinar (1992-1994)",
      "symbol": "BAD"
    },
    "BAM": {
      "displayName": "Bosnien und Herzegowina Konvertierbare Mark",
      "displayName-count-one": "Bosnien und Herzegowina Konvertierbare Mark",
      "displayName-count-other": "Bosnien und Herzegowina Konvertierbare Mark",
      "symbol": "BAM",
      "symbol-alt-narrow": "KM"
    },
    "BAN": {
      "displayName": "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
      "displayName-count-one": "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
      "displayName-count-other": "Bosnien und Herzegowina Neue Dinar (1994-1997)",
      "symbol": "BAN"
    },
    "BBD": {
      "displayName": "Barbados-Dollar",
      "displayName-count-one": "Barbados-Dollar",
      "displayName-count-other": "Barbados-Dollar",
      "symbol": "BBD",
      "symbol-alt-narrow": "$"
    },
    "BDT": {
      "displayName": "Bangladesch-Taka",
      "displayName-count-one": "Bangladesch-Taka",
      "displayName-count-other": "Bangladesch-Taka",
      "symbol": "BDT",
      "symbol-alt-narrow": "টাকা"
    },
    "BEC": {
      "displayName": "Belgischer Franc (konvertibel)",
      "displayName-count-one": "Belgischer Franc (konvertibel)",
      "displayName-count-other": "Belgische Franc (konvertibel)",

```

```

        "symbol": "BEC"
    },
    "BEF": {
        "displayName": "Belgischer Franc",
        "displayName-count-one": "Belgischer Franc",
        "displayName-count-other": "Belgische Franc",
        "symbol": "BEF"
    },
    "BEL": {
        "displayName": "Belgischer Finanz-Franc",
        "displayName-count-one": "Belgischer Finanz-Franc",
        "displayName-count-other": "Belgische Finanz-Franc",
        "symbol": "BEL"
    },
    "BGL": {
        "displayName": "Bulgarische Lew (1962-1999)",
        "displayName-count-one": "Bulgarische Lew (1962-1999)",
        "displayName-count-other": "Bulgarische Lew (1962-1999)",
        "symbol": "BGL"
    },
    "BGM": {
        "displayName": "Bulgarischer Lew (1952-1962)",
        "displayName-count-one": "Bulgarischer Lew (1952-1962)",
        "displayName-count-other": "Bulgarische Lew (1952-1962)",
        "symbol": "BGK"
    },
    "BGN": {
        "displayName": "Bulgarischer Lew",
        "displayName-count-one": "Bulgarischer Lew",
        "displayName-count-other": "Bulgarische Lew",
        "symbol": "BGN"
    },
    "BGO": {
        "displayName": "Bulgarischer Lew (1879-1952)",
        "displayName-count-one": "Bulgarischer Lew (1879-1952)",
        "displayName-count-other": "Bulgarische Lew (1879-1952)",
        "symbol": "BGJ"
    },
    "BHD": {
        "displayName": "Bahrain-Dinar",
        "displayName-count-one": "Bahrain-Dinar",
        "displayName-count-other": "Bahrain-Dinar",
        "symbol": "BHD"
    },
    "BIF": {
        "displayName": "Burundi-Franc",
        "displayName-count-one": "Burundi-Franc",
        "displayName-count-other": "Burundi-Francs",
        "symbol": "BIF"
    },
    "BMD": {
        "displayName": "Bermuda-Dollar",
        "displayName-count-one": "Bermuda-Dollar",
        "displayName-count-other": "Bermuda-Dollar",
        "symbol": "BMD",
        "symbol-alt-narrow": "$"
    },
    },

```

```

    "BND": {
      "displayName": "Brunei-Dollar",
      "displayName-count-one": "Brunei-Dollar",
      "displayName-count-other": "Brunei-Dollar",
      "symbol": "BND",
      "symbol-alt-narrow": "$"
    },
    "BOB": {
      "displayName": "Bolivanischer Boliviano",
      "displayName-count-one": "Bolivanischer Boliviano",
      "displayName-count-other": "Bolivianische Bolivianos",
      "symbol": "BOB",
      "symbol-alt-narrow": "Bs"
    },
    "BOL": {
      "displayName": "Bolivianischer Boliviano (1863-1963)",
      "displayName-count-one": "Bolivianischer Boliviano (1863-1963)",
      "displayName-count-other": "Bolivianische Bolivianos (1863-
1963)",
      "symbol": "BOL"
    },
    "BOP": {
      "displayName": "Bolivianischer Peso",
      "displayName-count-one": "Bolivianischer Peso",
      "displayName-count-other": "Bolivianische Peso",
      "symbol": "BOP"
    },
    "BOV": {
      "displayName": "Boliviansiche Mvdol",
      "displayName-count-one": "Boliviansiche Mvdol",
      "displayName-count-other": "Bolivianische Mvdol",
      "symbol": "BOV"
    },
    "BRB": {
      "displayName": "Brasilianischer Cruzeiro Novo (1967-1986)",
      "displayName-count-one": "Brasilianischer Cruzeiro Novo (1967-
1986)",
      "displayName-count-other": "Brasilianische Cruzeiro Novo (1967-
1986)",
      "symbol": "BRB"
    },
    "BRC": {
      "displayName": "Brasilianischer Cruzado (1986-1989)",
      "displayName-count-one": "Brasilianischer Cruzado (1986-1989)",
      "displayName-count-other": "Brasilianische Cruzado (1986-1989)",
      "symbol": "BRC"
    },
    "BRE": {
      "displayName": "Brasilianischer Cruzeiro (1990-1993)",
      "displayName-count-one": "Brasilianischer Cruzeiro (1990-1993)",
      "displayName-count-other": "Brasilianische Cruzeiro (1990-
1993)",
      "symbol": "BRE"
    },
    "BRL": {
      "displayName": "Brasilianischer Real",
      "displayName-count-one": "Brasilianischer Real",

```

```

        "displayName-count-other": "Brasilianische Real",
        "symbol": "R$",
        "symbol-alt-narrow": "R$"
    },
    "BRN": {
        "displayName": "Brasilianischer Cruzado Novo (1989-1990)",
        "displayName-count-one": "Brasilianischer Cruzado Novo (1989-
1990)",
        "displayName-count-other": "Brasilianische Cruzado Novo (1989-
1990)",
        "symbol": "BRN"
    },
    "BRR": {
        "displayName": "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-one": "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-other": "Brasilianische Cruzeiro (1993-
1994)",
        "symbol": "BRR"
    },
    "BRZ": {
        "displayName": "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-one": "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-other": "Brasilianischer Cruzeiro (1942-
1967)",
        "symbol": "BRZ"
    },
    "BSD": {
        "displayName": "Bahamas-Dollar",
        "displayName-count-one": "Bahamas-Dollar",
        "displayName-count-other": "Bahamas-Dollar",
        "symbol": "BSD",
        "symbol-alt-narrow": "$"
    },
    "BTN": {
        "displayName": "Bhutan-Ngultrum",
        "displayName-count-one": "Bhutan-Ngultrum",
        "displayName-count-other": "Bhutan-Ngultrum",
        "symbol": "BTN"
    },
    "BUK": {
        "displayName": "Birmanischer Kyat",
        "displayName-count-one": "Birmanischer Kyat",
        "displayName-count-other": "Birmanische Kyat",
        "symbol": "BUK"
    },
    "BWP": {
        "displayName": "Botswanischer Pula",
        "displayName-count-one": "Botswanischer Pula",
        "displayName-count-other": "Botswanische Pula",
        "symbol": "BWP",
        "symbol-alt-narrow": "P"
    },
    "BYB": {
        "displayName": "Belarus-Rubel (1994-1999)",
        "displayName-count-one": "Belarus-Rubel (1994-1999)",
        "displayName-count-other": "Belarus-Rubel (1994-1999)",
        "symbol": "BYB"
    }

```

```

    },
    "BYN": {
      "displayName": "Weißrussischer Rubel",
      "displayName-count-one": "Weißrussischer Rubel",
      "displayName-count-other": "Weißrussische Rubel",
      "symbol": "BYN",
      "symbol-alt-narrow": "p."
    },
    "BYR": {
      "displayName": "Weißrussischer Rubel (2000-2016)",
      "displayName-count-one": "Weißrussischer Rubel (2000-2016)",
      "displayName-count-other": "Weißrussische Rubel (2000-2016)",
      "symbol": "BYR"
    },
    "BZD": {
      "displayName": "Belize-Dollar",
      "displayName-count-one": "Belize-Dollar",
      "displayName-count-other": "Belize-Dollar",
      "symbol": "BZD",
      "symbol-alt-narrow": "$"
    },
    "CAD": {
      "displayName": "Kanadischer Dollar",
      "displayName-count-one": "Kanadischer Dollar",
      "displayName-count-other": "Kanadische Dollar",
      "symbol": "CA$",
      "symbol-alt-narrow": "$"
    },
    "CDF": {
      "displayName": "Kongo-Franc",
      "displayName-count-one": "Kongo-Franc",
      "displayName-count-other": "Kongo-Francis",
      "symbol": "CDF"
    },
    "CHE": {
      "displayName": "WIR-Euro",
      "displayName-count-one": "WIR-Euro",
      "displayName-count-other": "WIR-Euro",
      "symbol": "CHE"
    },
    "CHF": {
      "displayName": "Schweizer Franken",
      "displayName-count-one": "Schweizer Franken",
      "displayName-count-other": "Schweizer Franken",
      "symbol": "CHF"
    },
    "CHW": {
      "displayName": "WIR Franken",
      "displayName-count-one": "WIR Franken",
      "displayName-count-other": "WIR Franken",
      "symbol": "CHW"
    },
    "CLE": {
      "displayName": "Chilenischer Escudo",
      "displayName-count-one": "Chilenischer Escudo",
      "displayName-count-other": "Chilenische Escudo",
      "symbol": "CLE"
    }
  }

```



```

    },
    "CLF": {
      "displayName": "Chilenische Unidades de Fomento",
      "displayName-count-one": "Chilenische Unidades de Fomento",
      "displayName-count-other": "Chilenische Unidades de Fomento",
      "symbol": "CLF"
    },
    "CLP": {
      "displayName": "Chilenischer Peso",
      "displayName-count-one": "Chilenischer Peso",
      "displayName-count-other": "Chilenische Pesos",
      "symbol": "CLP",
      "symbol-alt-narrow": "$"
    },
    "CNX": {
      "displayName": "Dollar der Chinesischen Volksbank",
      "displayName-count-one": "Dollar der Chinesischen Volksbank",
      "displayName-count-other": "Dollar der Chinesischen Volksbank",
      "symbol": "CNX"
    },
    "CNY": {
      "displayName": "Renminbi Yuan",
      "displayName-count-one": "Chinesischer Yuan",
      "displayName-count-other": "Renminbi Yuan",
      "symbol": "CN¥",
      "symbol-alt-narrow": "¥"
    },
    "COP": {
      "displayName": "Kolumbianischer Peso",
      "displayName-count-one": "Kolumbianischer Peso",
      "displayName-count-other": "Kolumbianische Pesos",
      "symbol": "COP",
      "symbol-alt-narrow": "$"
    },
    "COU": {
      "displayName": "Kolumbianische Unidades de valor real",
      "displayName-count-one": "Kolumbianische Unidad de valor real",
      "displayName-count-other": "Kolumbianische Unidades de valor
real",
      "symbol": "COU"
    },
    "CRC": {
      "displayName": "Costa-Rica-Colón",
      "displayName-count-one": "Costa-Rica-Colón",
      "displayName-count-other": "Costa-Rica-Colón",
      "symbol": "CRC",
      "symbol-alt-narrow": "₡"
    },
    "CSD": {
      "displayName": "Serbischer Dinar (2002-2006)",
      "displayName-count-one": "Serbischer Dinar (2002-2006)",
      "displayName-count-other": "Serbische Dinar (2002-2006)",
      "symbol": "CSD"
    },
    "CSK": {
      "displayName": "Tschechoslowakische Krone",
      "displayName-count-one": "Tschechoslowakische Kronen",

```

```

        "displayName-count-other": "Tschechoslowakische Kronen",
        "symbol": "CSK"
    },
    "CUC": {
        "displayName": "Kubanischer Peso (konvertibel)",
        "displayName-count-one": "Kubanischer Peso (konvertibel)",
        "displayName-count-other": "Kubanische Pesos (konvertibel)",
        "symbol": "CUC",
        "symbol-alt-narrow": "Cub$"
    },
    "CUP": {
        "displayName": "Kubanischer Peso",
        "displayName-count-one": "Kubanischer Peso",
        "displayName-count-other": "Kubanische Pesos",
        "symbol": "CUP",
        "symbol-alt-narrow": "$"
    },
    "CVE": {
        "displayName": "Cabo-Verde-Escudo",
        "displayName-count-one": "Cabo-Verde-Escudo",
        "displayName-count-other": "Cabo-Verde-Escudos",
        "symbol": "CVE"
    },
    "CYP": {
        "displayName": "Zypern-Pfund",
        "displayName-count-one": "Zypern Pfund",
        "displayName-count-other": "Zypern Pfund",
        "symbol": "CYP"
    },
    "CZK": {
        "displayName": "Tschechische Krone",
        "displayName-count-one": "Tschechische Krone",
        "displayName-count-other": "Tschechische Kronen",
        "symbol": "CZK",
        "symbol-alt-narrow": "Kč"
    },
    "DDM": {
        "displayName": "Mark der DDR",
        "displayName-count-one": "Mark der DDR",
        "displayName-count-other": "Mark der DDR",
        "symbol": "DDM"
    },
    "DEM": {
        "displayName": "Deutsche Mark",
        "displayName-count-one": "Deutsche Mark",
        "displayName-count-other": "Deutsche Mark",
        "symbol": "DM"
    },
    "DJF": {
        "displayName": "Dschibuti-Franc",
        "displayName-count-one": "Dschibuti-Franc",
        "displayName-count-other": "Dschibuti-Franc",
        "symbol": "DJF"
    },
    "DKK": {
        "displayName": "Dänische Krone",
        "displayName-count-one": "Dänische Krone",

```

```

        "displayName-count-other": "Dänische Kronen",
        "symbol": "DKK",
        "symbol-alt-narrow": "kr"
    },
    "DOP": {
        "displayName": "Dominikanischer Peso",
        "displayName-count-one": "Dominikanischer Peso",
        "displayName-count-other": "Dominikanische Pesos",
        "symbol": "DOP",
        "symbol-alt-narrow": "$"
    },
    "DZD": {
        "displayName": "Algerischer Dinar",
        "displayName-count-one": "Algerischer Dinar",
        "displayName-count-other": "Algerische Dinar",
        "symbol": "DZD"
    },
    "ECS": {
        "displayName": "Ecuadorianischer Sucre",
        "displayName-count-one": "Ecuadorianischer Sucre",
        "displayName-count-other": "Ecuadorianische Sucre",
        "symbol": "ECS"
    },
    "ECV": {
        "displayName": "Verrechnungseinheit für Ecuador",
        "displayName-count-one": "Verrechnungseinheiten für Ecuador",
        "displayName-count-other": "Verrechnungseinheiten für Ecuador",
        "symbol": "ECV"
    },
    "EEK": {
        "displayName": "Estnische Krone",
        "displayName-count-one": "Estnische Krone",
        "displayName-count-other": "Estnische Kronen",
        "symbol": "EEK"
    },
    "EGP": {
        "displayName": "Ägyptisches Pfund",
        "displayName-count-one": "Ägyptisches Pfund",
        "displayName-count-other": "Ägyptische Pfund",
        "symbol": "EGP",
        "symbol-alt-narrow": "E£"
    },
    "ERN": {
        "displayName": "Eritreischer Nakfa",
        "displayName-count-one": "Eritreischer Nakfa",
        "displayName-count-other": "Eritreische Nakfa",
        "symbol": "ERN"
    },
    "ESA": {
        "displayName": "Spanische Peseta (A-Konten)",
        "displayName-count-one": "Spanische Peseta (A-Konten)",
        "displayName-count-other": "Spanische Peseten (A-Konten)",
        "symbol": "ESA"
    },
    "ESB": {
        "displayName": "Spanische Peseta (konvertibel)",
        "displayName-count-one": "Spanische Peseta (konvertibel)",

```

```

        "displayName-count-other": "Spanische Peseten (konvertibel)",
        "symbol": "ESB"
    },
    "ESP": {
        "displayName": "Spanische Peseta",
        "displayName-count-one": "Spanische Peseta",
        "displayName-count-other": "Spanische Peseten",
        "symbol": "ESP",
        "symbol-alt-narrow": "₧"
    },
    "ETB": {
        "displayName": "Äthiopischer Birr",
        "displayName-count-one": "Äthiopischer Birr",
        "displayName-count-other": "Äthiopische Birr",
        "symbol": "ETB"
    },
    "EUR": {
        "displayName": "Euro",
        "displayName-count-one": "Euro",
        "displayName-count-other": "Euro",
        "symbol": "€",
        "symbol-alt-narrow": "€"
    },
    "FIM": {
        "displayName": "Finnische Mark",
        "displayName-count-one": "Finnische Mark",
        "displayName-count-other": "Finnische Mark",
        "symbol": "FIM"
    },
    "FJD": {
        "displayName": "Fidschi-Dollar",
        "displayName-count-one": "Fidschi-Dollar",
        "displayName-count-other": "Fidschi-Dollar",
        "symbol": "FJD",
        "symbol-alt-narrow": "$"
    },
    "FKP": {
        "displayName": "Falkland-Pfund",
        "displayName-count-one": "Falkland-Pfund",
        "displayName-count-other": "Falkland-Pfund",
        "symbol": "FKP",
        "symbol-alt-narrow": "Fl£"
    },
    "FRF": {
        "displayName": "Französischer Franc",
        "displayName-count-one": "Französischer Franc",
        "displayName-count-other": "Französische Franc",
        "symbol": "FRF"
    },
    "GBP": {
        "displayName": "Britisches Pfund",
        "displayName-count-one": "Britisches Pfund",
        "displayName-count-other": "Britische Pfund",
        "symbol": "£",
        "symbol-alt-narrow": "£"
    },
    "GEK": {

```

```

        "displayName": "Georgischer Kupon Larit",
        "displayName-count-one": "Georgischer Kupon Larit",
        "displayName-count-other": "Georgische Kupon Larit",
        "symbol": "GEK"
    },
    "GEL": {
        "displayName": "Georgischer Lari",
        "displayName-count-one": "Georgischer Lari",
        "displayName-count-other": "Georgische Lari",
        "symbol": "GEL",
        "symbol-alt-narrow": "ლ",
        "symbol-alt-variant": "ლ"
    },
    "GHC": {
        "displayName": "Ghanaischer Cedi (1979-2007)",
        "displayName-count-one": "Ghanaischer Cedi (1979-2007)",
        "displayName-count-other": "Ghanaische Cedi (1979-2007)",
        "symbol": "GHC"
    },
    "GHS": {
        "displayName": "Ghanaischer Cedi",
        "displayName-count-one": "Ghanaischer Cedi",
        "displayName-count-other": "Ghanaische Cedi",
        "symbol": "GHS"
    },
    "GIP": {
        "displayName": "Gibraltar-Pfund",
        "displayName-count-one": "Gibraltar-Pfund",
        "displayName-count-other": "Gibraltar Pfund",
        "symbol": "GIP",
        "symbol-alt-narrow": "£"
    },
    "GMD": {
        "displayName": "Gambia-Dalasi",
        "displayName-count-one": "Gambia-Dalasi",
        "displayName-count-other": "Gambia-Dalasi",
        "symbol": "GMD"
    },
    "GNF": {
        "displayName": "Guinea-Franc",
        "displayName-count-one": "Guinea-Franc",
        "displayName-count-other": "Guinea-Franc",
        "symbol": "GNF",
        "symbol-alt-narrow": "F.G."
    },
    "GNS": {
        "displayName": "Guineischer Syli",
        "displayName-count-one": "Guineischer Syli",
        "displayName-count-other": "Guineische Syli",
        "symbol": "GNS"
    },
    "GQE": {
        "displayName": "Äquatorialguinea-Ekwele",
        "displayName-count-one": "Äquatorialguinea-Ekwele",
        "displayName-count-other": "Äquatorialguinea-Ekwele",
        "symbol": "GQE"
    },
    },

```

```

"GRD": {
  "displayName": "Griechische Drachme",
  "displayName-count-one": "Griechische Drachme",
  "displayName-count-other": "Griechische Drachmen",
  "symbol": "GRD"
},
"GTQ": {
  "displayName": "Guatemaltekischer Quetzal",
  "displayName-count-one": "Guatemaltekischer Quetzal",
  "displayName-count-other": "Guatemaltekische Quetzales",
  "symbol": "GTQ",
  "symbol-alt-narrow": "Q"
},
"GWE": {
  "displayName": "Portugiesisch Guinea Escudo",
  "displayName-count-one": "Portugiesisch Guinea Escudo",
  "displayName-count-other": "Portugiesisch Guinea Escudo",
  "symbol": "GWE"
},
"GWP": {
  "displayName": "Guinea-Bissau Peso",
  "displayName-count-one": "Guinea-Bissau Peso",
  "displayName-count-other": "Guinea-Bissau Pesos",
  "symbol": "GWP"
},
"GYD": {
  "displayName": "Guyana-Dollar",
  "displayName-count-one": "Guyana-Dollar",
  "displayName-count-other": "Guyana-Dollar",
  "symbol": "GYD",
  "symbol-alt-narrow": "$"
},
"HKD": {
  "displayName": "Hongkong-Dollar",
  "displayName-count-one": "Hongkong-Dollar",
  "displayName-count-other": "Hongkong-Dollar",
  "symbol": "HK$",
  "symbol-alt-narrow": "$"
},
"HNL": {
  "displayName": "Honduras-Lempira",
  "displayName-count-one": "Honduras-Lempira",
  "displayName-count-other": "Honduras-Lempira",
  "symbol": "HNL",
  "symbol-alt-narrow": "L"
},
"HRD": {
  "displayName": "Kroatischer Dinar",
  "displayName-count-one": "Kroatischer Dinar",
  "displayName-count-other": "Kroatische Dinar",
  "symbol": "HRD"
},
"HRK": {
  "displayName": "Kroatischer Kuna",
  "displayName-count-one": "Kroatischer Kuna",
  "displayName-count-other": "Kroatische Kuna",
  "symbol": "HRK",

```

```

        "symbol-alt-narrow": "kn"
    },
    "HTG": {
        "displayName": "Haitianische Gourde",
        "displayName-count-one": "Haitianische Gourde",
        "displayName-count-other": "Haitianische Gourdes",
        "symbol": "HTG"
    },
    "HUF": {
        "displayName": "Ungarischer Forint",
        "displayName-count-one": "Ungarischer Forint",
        "displayName-count-other": "Ungarische Forint",
        "symbol": "HUF",
        "symbol-alt-narrow": "Ft"
    },
    "IDR": {
        "displayName": "Indonesische Rupiah",
        "displayName-count-one": "Indonesische Rupiah",
        "displayName-count-other": "Indonesische Rupiah",
        "symbol": "IDR",
        "symbol-alt-narrow": "Rp"
    },
    "IEP": {
        "displayName": "Irisches Pfund",
        "displayName-count-one": "Irisches Pfund",
        "displayName-count-other": "Irische Pfund",
        "symbol": "IEP"
    },
    "ILP": {
        "displayName": "Israelisches Pfund",
        "displayName-count-one": "Israelisches Pfund",
        "displayName-count-other": "Israelische Pfund",
        "symbol": "ILP"
    },
    "ILR": {
        "displayName": "Israelischer Schekel (1980-1985)",
        "displayName-count-one": "Israelischer Schekel (1980-1985)",
        "displayName-count-other": "Israelische Schekel (1980-1985)"
    },
    "ILS": {
        "displayName": "Israelischer Neuer Schekel",
        "displayName-count-one": "Israelischer Neuer Schekel",
        "displayName-count-other": "Israelische Neue Schekel",
        "symbol": "₪",
        "symbol-alt-narrow": "₪"
    },
    "INR": {
        "displayName": "Indische Rupie",
        "displayName-count-one": "Indische Rupie",
        "displayName-count-other": "Indische Rupien",
        "symbol": "₹",
        "symbol-alt-narrow": "₹"
    },
    "IQD": {
        "displayName": "Irakischer Dinar",
        "displayName-count-one": "Irakischer Dinar",
        "displayName-count-other": "Irakische Dinar",

```

```

        "symbol": "IQD"
    },
    "IRR": {
        "displayName": "Iranischer Rial",
        "displayName-count-one": "Iranischer Rial",
        "displayName-count-other": "Iranische Rial",
        "symbol": "IRR"
    },
    "ISJ": {
        "displayName": "Isländische Krone (1918-1981)",
        "displayName-count-one": "Isländische Krone (1918-1981)",
        "displayName-count-other": "Isländische Kronen (1918-1981)"
    },
    "ISK": {
        "displayName": "Isländische Krone",
        "displayName-count-one": "Isländische Krone",
        "displayName-count-other": "Isländische Kronen",
        "symbol": "ISK",
        "symbol-alt-narrow": "kr"
    },
    "ITL": {
        "displayName": "Italienische Lira",
        "displayName-count-one": "Italienische Lira",
        "displayName-count-other": "Italienische Lire",
        "symbol": "ITL"
    },
    "JMD": {
        "displayName": "Jamaika-Dollar",
        "displayName-count-one": "Jamaika-Dollar",
        "displayName-count-other": "Jamaika-Dollar",
        "symbol": "JMD",
        "symbol-alt-narrow": "$"
    },
    "JOD": {
        "displayName": "Jordanischer Dinar",
        "displayName-count-one": "Jordanischer Dinar",
        "displayName-count-other": "Jordanische Dinar",
        "symbol": "JOD"
    },
    "JPY": {
        "displayName": "Japanischer Yen",
        "displayName-count-one": "Japanischer Yen",
        "displayName-count-other": "Japanische Yen",
        "symbol": "¥",
        "symbol-alt-narrow": "¥"
    },
    "KES": {
        "displayName": "Kenia-Schilling",
        "displayName-count-one": "Kenia-Schilling",
        "displayName-count-other": "Kenia-Schilling",
        "symbol": "KES"
    },
    "KGS": {
        "displayName": "Kirgisischer Som",
        "displayName-count-one": "Kirgisischer Som",
        "displayName-count-other": "Kirgisische Som",
        "symbol": "KGS"
    }

```



```

    },
    "KHR": {
      "displayName": "Kambodschanischer Riel",
      "displayName-count-one": "Kambodschanischer Riel",
      "displayName-count-other": "Kambodschanische Riel",
      "symbol": "KHR",
      "symbol-alt-narrow": "៛"
    },
    "KMF": {
      "displayName": "Komoren-Franc",
      "displayName-count-one": "Komoren-Franc",
      "displayName-count-other": "Komoren-Francis",
      "symbol": "KMF",
      "symbol-alt-narrow": "FC"
    },
    "KPW": {
      "displayName": "Nordkoreanischer Won",
      "displayName-count-one": "Nordkoreanischer Won",
      "displayName-count-other": "Nordkoreanische Won",
      "symbol": "KPW",
      "symbol-alt-narrow": "₩"
    },
    "KRH": {
      "displayName": "Südkoreanischer Hwan (1953-1962)",
      "displayName-count-one": "Südkoreanischer Hwan (1953-1962)",
      "displayName-count-other": "Südkoreanischer Hwan (1953-1962)",
      "symbol": "KRH"
    },
    "KRO": {
      "displayName": "Südkoreanischer Won (1945-1953)",
      "displayName-count-one": "Südkoreanischer Won (1945-1953)",
      "displayName-count-other": "Südkoreanischer Won (1945-1953)",
      "symbol": "KRO"
    },
    "KRW": {
      "displayName": "Südkoreanischer Won",
      "displayName-count-one": "Südkoreanischer Won",
      "displayName-count-other": "Südkoreanische Won",
      "symbol": "₩",
      "symbol-alt-narrow": "₩"
    },
    "KWD": {
      "displayName": "Kuwait-Dinar",
      "displayName-count-one": "Kuwait-Dinar",
      "displayName-count-other": "Kuwait-Dinar",
      "symbol": "KWD"
    },
    "KYD": {
      "displayName": "Kaiman-Dollar",
      "displayName-count-one": "Kaiman-Dollar",
      "displayName-count-other": "Kaiman-Dollar",
      "symbol": "KYD",
      "symbol-alt-narrow": "$"
    },
    "KZT": {
      "displayName": "Kasachischer Tenge",

```

```

        "displayName-count-one": "Kasachischer Tenge",
        "displayName-count-other": "Kasachische Tenge",
        "symbol": "KZT",
        "symbol-alt-narrow": "Т"
    },
    "LAK": {
        "displayName": "Laotischer Kip",
        "displayName-count-one": "Laotischer Kip",
        "displayName-count-other": "Laotische Kip",
        "symbol": "LAK",
        "symbol-alt-narrow": "₭"
    },
    "LBP": {
        "displayName": "Libanesisches Pfund",
        "displayName-count-one": "Libanesisches Pfund",
        "displayName-count-other": "Libanesische Pfund",
        "symbol": "LBP",
        "symbol-alt-narrow": "L£"
    },
    "LKR": {
        "displayName": "Sri-Lanka-Rupie",
        "displayName-count-one": "Sri-Lanka-Rupie",
        "displayName-count-other": "Sri-Lanka-Rupien",
        "symbol": "LKR",
        "symbol-alt-narrow": "Rs"
    },
    "LRD": {
        "displayName": "Liberianischer Dollar",
        "displayName-count-one": "Liberianischer Dollar",
        "displayName-count-other": "Liberianische Dollar",
        "symbol": "LRD",
        "symbol-alt-narrow": "$"
    },
    "LSL": {
        "displayName": "Loti",
        "displayName-count-one": "Loti",
        "displayName-count-other": "Loti",
        "symbol": "LSL"
    },
    "LTL": {
        "displayName": "Litauischer Litas",
        "displayName-count-one": "Litauischer Litas",
        "displayName-count-other": "Litauische Litas",
        "symbol": "LTL",
        "symbol-alt-narrow": "Lt"
    },
    "LTT": {
        "displayName": "Litauischer Talonas",
        "displayName-count-one": "Litauische Talonas",
        "displayName-count-other": "Litauische Talonas",
        "symbol": "LTT"
    },
    "LUC": {
        "displayName": "Luxemburgischer Franc (konvertibel)",
        "displayName-count-one": "Luxemburgische Franc (konvertibel)",
        "displayName-count-other": "Luxemburgische Franc (konvertibel)",
        "symbol": "LUC"
    }

```

```

    },
    "LUF": {
      "displayName": "Luxemburgischer Franc",
      "displayName-count-one": "Luxemburgische Franc",
      "displayName-count-other": "Luxemburgische Franc",
      "symbol": "LUF"
    },
    "LUL": {
      "displayName": "Luxemburgischer Finanz-Franc",
      "displayName-count-one": "Luxemburgische Finanz-Franc",
      "displayName-count-other": "Luxemburgische Finanz-Franc",
      "symbol": "LUL"
    },
    "LVL": {
      "displayName": "Lettischer Lats",
      "displayName-count-one": "Lettischer Lats",
      "displayName-count-other": "Lettische Lats",
      "symbol": "LVL",
      "symbol-alt-narrow": "Ls"
    },
    "LVR": {
      "displayName": "Lettischer Rubel",
      "displayName-count-one": "Lettische Rubel",
      "displayName-count-other": "Lettische Rubel",
      "symbol": "LVR"
    },
    "LYD": {
      "displayName": "Libyscher Dinar",
      "displayName-count-one": "Libyscher Dinar",
      "displayName-count-other": "Libysche Dinar",
      "symbol": "LYD"
    },
    "MAD": {
      "displayName": "Marokkanischer Dirham",
      "displayName-count-one": "Marokkanischer Dirham",
      "displayName-count-other": "Marokkanische Dirham",
      "symbol": "MAD"
    },
    "MAF": {
      "displayName": "Marokkanischer Franc",
      "displayName-count-one": "Marokkanische Franc",
      "displayName-count-other": "Marokkanische Franc",
      "symbol": "MAF"
    },
    "MCF": {
      "displayName": "Monegassischer Franc",
      "displayName-count-one": "Monegassischer Franc",
      "displayName-count-other": "Monegassische Franc",
      "symbol": "MCF"
    },
    "MDC": {
      "displayName": "Moldau-Cupon",
      "displayName-count-one": "Moldau-Cupon",
      "displayName-count-other": "Moldau-Cupon",
      "symbol": "MDC"
    },
    "MDL": {

```

```

        "displayName": "Moldau-Leu",
        "displayName-count-one": "Moldau-Leu",
        "displayName-count-other": "Moldau-Leu",
        "symbol": "MDL"
    },
    "MGA": {
        "displayName": "Madagaskar-Ariary",
        "displayName-count-one": "Madagaskar-Ariary",
        "displayName-count-other": "Madagaskar-Ariary",
        "symbol": "MGA",
        "symbol-alt-narrow": "Ar"
    },
    "MGF": {
        "displayName": "Madagaskar-Franc",
        "displayName-count-one": "Madagaskar-Franc",
        "displayName-count-other": "Madagaskar-Franc",
        "symbol": "MGF"
    },
    "MKD": {
        "displayName": "Mazedonischer Denar",
        "displayName-count-one": "Mazedonischer Denar",
        "displayName-count-other": "Mazedonische Denari",
        "symbol": "MKD"
    },
    "MKN": {
        "displayName": "Mazedonischer Denar (1992-1993)",
        "displayName-count-one": "Mazedonischer Denar (1992-1993)",
        "displayName-count-other": "Mazedonische Denar (1992-1993)",
        "symbol": "MKN"
    },
    "MLF": {
        "displayName": "Malischer Franc",
        "displayName-count-one": "Malische Franc",
        "displayName-count-other": "Malische Franc",
        "symbol": "MLF"
    },
    "MMK": {
        "displayName": "Myanmarischer Kyat",
        "displayName-count-one": "Myanmarischer Kyat",
        "displayName-count-other": "Myanmarische Kyat",
        "symbol": "MMK",
        "symbol-alt-narrow": "K"
    },
    "MNT": {
        "displayName": "Mongolischer Tögrög",
        "displayName-count-one": "Mongolischer Tögrög",
        "displayName-count-other": "Mongolische Tögrög",
        "symbol": "MNT",
        "symbol-alt-narrow": "₮"
    },
    "MOP": {
        "displayName": "Macao-Pataca",
        "displayName-count-one": "Macao-Pataca",
        "displayName-count-other": "Macao-Pataca",
        "symbol": "MOP"
    },
    "MRO": {

```

```

        "displayName": "Mauretanischer Ouguiya",
        "displayName-count-one": "Mauretanischer Ouguiya",
        "displayName-count-other": "Mauretansiche Ouguiya",
        "symbol": "MRO"
    },
    "MTL": {
        "displayName": "Maltesische Lira",
        "displayName-count-one": "Maltesische Lira",
        "displayName-count-other": "Maltesische Lira",
        "symbol": "MTL"
    },
    "MTP": {
        "displayName": "Maltesisches Pfund",
        "displayName-count-one": "Maltesische Pfund",
        "displayName-count-other": "Maltesische Pfund",
        "symbol": "MTP"
    },
    "MUR": {
        "displayName": "Mauritius-Rupie",
        "displayName-count-one": "Mauritius-Rupie",
        "displayName-count-other": "Mauritius-Rupien",
        "symbol": "MUR",
        "symbol-alt-narrow": "Rs"
    },
    "MVP": {
        "displayName": "Malediven-Rupie (alt)",
        "displayName-count-one": "Malediven-Rupie (alt)",
        "displayName-count-other": "Malediven-Rupien (alt)"
    },
    "MVR": {
        "displayName": "Malediven-Rufiyaa",
        "displayName-count-one": "Malediven-Rufiyaa",
        "displayName-count-other": "Malediven-Rupien",
        "symbol": "MVR"
    },
    "MWK": {
        "displayName": "Malawi-Kwacha",
        "displayName-count-one": "Malawi-Kwacha",
        "displayName-count-other": "Malawi-Kwacha",
        "symbol": "MWK"
    },
    "MXN": {
        "displayName": "Mexikanischer Peso",
        "displayName-count-one": "Mexikanischer Peso",
        "displayName-count-other": "Mexikanische Pesos",
        "symbol": "MX$",
        "symbol-alt-narrow": "$"
    },
    "MXP": {
        "displayName": "Mexikanischer Silber-Peso (1861-1992)",
        "displayName-count-one": "Mexikanische Silber-Peso (1861-1992)",
        "displayName-count-other": "Mexikanische Silber-Pesos (1861-
1992)",
        "symbol": "MXP"
    },
    "MXV": {
        "displayName": "Mexicanischer Unidad de Inversion (UDI)",

```

```

        "displayName-count-one": "Mexicanischer Unidad de Inversion
(UDI) ",
        "displayName-count-other": "Mexikanische Unidad de Inversion
(UDI) ",
        "symbol": "MXV"
    },
    "MYR": {
        "displayName": "Malaysischer Ringgit",
        "displayName-count-one": "Malaysischer Ringgit",
        "displayName-count-other": "Malaysische Ringgit",
        "symbol": "MYR",
        "symbol-alt-narrow": "RM"
    },
    "MZE": {
        "displayName": "Mosambikanischer Escudo",
        "displayName-count-one": "Mozambikanische Escudo",
        "displayName-count-other": "Mozambikanische Escudo",
        "symbol": "MZE"
    },
    "MZM": {
        "displayName": "Mosambikanischer Metical (1980-2006)",
        "displayName-count-one": "Mosambikanischer Metical (1980-2006)",
        "displayName-count-other": "Mosambikanische Meticais (1980-
2006) ",
        "symbol": "MZM"
    },
    "MZN": {
        "displayName": "Mosambikanischer Metical",
        "displayName-count-one": "Mosambikanischer Metical",
        "displayName-count-other": "Mosambikanische Meticais",
        "symbol": "MZN"
    },
    "NAD": {
        "displayName": "Namibia-Dollar",
        "displayName-count-one": "Namibia-Dollar",
        "displayName-count-other": "Namibia-Dollar",
        "symbol": "NAD",
        "symbol-alt-narrow": "$"
    },
    "NGN": {
        "displayName": "Nigerianischer Naira",
        "displayName-count-one": "Nigerianischer Naira",
        "displayName-count-other": "Nigerianische Naira",
        "symbol": "NGN",
        "symbol-alt-narrow": "₦"
    },
    "NIC": {
        "displayName": "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-one": "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-other": "Nicaraguanische Córdoba (1988-
1991) ",
        "symbol": "NIC"
    },
    "NIO": {
        "displayName": "Nicaragua-Córdoba",
        "displayName-count-one": "Nicaragua-Córdoba",
        "displayName-count-other": "Nicaragua-Córdobas",

```

```

        "symbol": "NIO",
        "symbol-alt-narrow": "C$"
    },
    "NLG": {
        "displayName": "Niederländischer Gulden",
        "displayName-count-one": "Niederländischer Gulden",
        "displayName-count-other": "Niederländische Gulden",
        "symbol": "NLG"
    },
    "NOK": {
        "displayName": "Norwegische Krone",
        "displayName-count-one": "Norwegische Krone",
        "displayName-count-other": "Norwegische Kronen",
        "symbol": "NOK",
        "symbol-alt-narrow": "kr"
    },
    "NPR": {
        "displayName": "Nepalesische Rupie",
        "displayName-count-one": "Nepalesische Rupie",
        "displayName-count-other": "Nepalesische Rupien",
        "symbol": "NPR",
        "symbol-alt-narrow": "Rs"
    },
    "NZD": {
        "displayName": "Neuseeland-Dollar",
        "displayName-count-one": "Neuseeland-Dollar",
        "displayName-count-other": "Neuseeland-Dollar",
        "symbol": "NZ$",
        "symbol-alt-narrow": "$"
    },
    "OMR": {
        "displayName": "Omanischer Rial",
        "displayName-count-one": "Omanischer Rial",
        "displayName-count-other": "Omanische Rials",
        "symbol": "OMR"
    },
    "PAB": {
        "displayName": "Panamaischer Balboa",
        "displayName-count-one": "Panamaischer Balboa",
        "displayName-count-other": "Panamaische Balboas",
        "symbol": "PAB"
    },
    "PEI": {
        "displayName": "Peruanischer Inti",
        "displayName-count-one": "Peruanische Inti",
        "displayName-count-other": "Peruanische Inti",
        "symbol": "PEI"
    },
    "PEN": {
        "displayName": "Peruanischer Sol",
        "displayName-count-one": "Peruanischer Sol",
        "displayName-count-other": "Peruanische Sol",
        "symbol": "PEN"
    },
    "PES": {
        "displayName": "Peruanischer Sol (1863-1965)",
        "displayName-count-one": "Peruanischer Sol (1863-1965)",

```

```

        "displayName-count-other": "Peruanische Sol (1863-1965)",
        "symbol": "PES"
    },
    "PGK": {
        "displayName": "Papua-Neuguineischer Kina",
        "displayName-count-one": "Papua-Neuguineischer Kina",
        "displayName-count-other": "Papua-Neuguineische Kina",
        "symbol": "PGK"
    },
    "PHP": {
        "displayName": "Philippinischer Peso",
        "displayName-count-one": "Philippinischer Peso",
        "displayName-count-other": "Philippinische Pesos",
        "symbol": "PHP",
        "symbol-alt-narrow": "₱"
    },
    "PKR": {
        "displayName": "Pakistanische Rupie",
        "displayName-count-one": "Pakistanische Rupie",
        "displayName-count-other": "Pakistanische Rupien",
        "symbol": "PKR",
        "symbol-alt-narrow": "Rs"
    },
    "PLN": {
        "displayName": "Polnischer Złoty",
        "displayName-count-one": "Polnischer Złoty",
        "displayName-count-other": "Polnische Złoty",
        "symbol": "PLN",
        "symbol-alt-narrow": "zł"
    },
    "PLZ": {
        "displayName": "Polnischer Zloty (1950-1995)",
        "displayName-count-one": "Polnischer Zloty (1950-1995)",
        "displayName-count-other": "Polnische Zloty (1950-1995)",
        "symbol": "PLZ"
    },
    "PTE": {
        "displayName": "Portugiesischer Escudo",
        "displayName-count-one": "Portugiesische Escudo",
        "displayName-count-other": "Portugiesische Escudo",
        "symbol": "PTE"
    },
    "PYG": {
        "displayName": "Paraguayischer Guaraní",
        "displayName-count-one": "Paraguayischer Guaraní",
        "displayName-count-other": "Paraguayische Guaraníes",
        "symbol": "PYG",
        "symbol-alt-narrow": "₲"
    },
    "QAR": {
        "displayName": "Katar-Riyal",
        "displayName-count-one": "Katar-Riyal",
        "displayName-count-other": "Katar-Riyal",
        "symbol": "QAR"
    },
    "RHD": {
        "displayName": "Rhodesischer Dollar",

```



```

        "displayName-count-one": "Rhodesische Dollar",
        "displayName-count-other": "Rhodesische Dollar",
        "symbol": "RHD"
    },
    "ROL": {
        "displayName": "Rumänischer Leu (1952-2006)",
        "displayName-count-one": "Rumänischer Leu (1952-2006)",
        "displayName-count-other": "Rumänische Leu (1952-2006)",
        "symbol": "ROL"
    },
    "RON": {
        "displayName": "Rumänischer Leu",
        "displayName-count-one": "Rumänischer Leu",
        "displayName-count-other": "Rumänische Leu",
        "symbol": "RON",
        "symbol-alt-narrow": "L"
    },
    "RSD": {
        "displayName": "Serbischer Dinar",
        "displayName-count-one": "Serbischer Dinar",
        "displayName-count-other": "Serbische Dinaren",
        "symbol": "RSD"
    },
    "RUB": {
        "displayName": "Russischer Rubel",
        "displayName-count-one": "Russischer Rubel",
        "displayName-count-other": "Russische Rubel",
        "symbol": "RUB",
        "symbol-alt-narrow": "₽"
    },
    "RUR": {
        "displayName": "Russischer Rubel (1991-1998)",
        "displayName-count-one": "Russischer Rubel (1991-1998)",
        "displayName-count-other": "Russische Rubel (1991-1998)",
        "symbol": "RUR",
        "symbol-alt-narrow": "p."
    },
    "RWF": {
        "displayName": "Ruanda-Franc",
        "displayName-count-one": "Ruanda-Franc",
        "displayName-count-other": "Ruanda-Francs",
        "symbol": "RWF",
        "symbol-alt-narrow": "F.Rw"
    },
    "SAR": {
        "displayName": "Saudi-Rial",
        "displayName-count-one": "Saudi-Rial",
        "displayName-count-other": "Saudi-Rial",
        "symbol": "SAR"
    },
    "SBD": {
        "displayName": "Salomonen-Dollar",
        "displayName-count-one": "Salomonen-Dollar",
        "displayName-count-other": "Salomonen-Dollar",
        "symbol": "SBD",
        "symbol-alt-narrow": "$"
    },
    },

```

```

"SCR": {
  "displayName": "Seychellen-Rupie",
  "displayName-count-one": "Seychellen-Rupie",
  "displayName-count-other": "Seychellen-Rupien",
  "symbol": "SCR"
},
"SDD": {
  "displayName": "Sudanesischer Dinar (1992-2007)",
  "displayName-count-one": "Sudanesischer Dinar (1992-2007)",
  "displayName-count-other": "Sudanesische Dinar (1992-2007)",
  "symbol": "SDD"
},
"SDG": {
  "displayName": "Sudanesisches Pfund",
  "displayName-count-one": "Sudanesisches Pfund",
  "displayName-count-other": "Sudanesische Pfund",
  "symbol": "SDG"
},
"SDP": {
  "displayName": "Sudanesisches Pfund (1957-1998)",
  "displayName-count-one": "Sudanesisches Pfund (1957-1998)",
  "displayName-count-other": "Sudanesische Pfund (1957-1998)",
  "symbol": "SDP"
},
"SEK": {
  "displayName": "Schwedische Krone",
  "displayName-count-one": "Schwedische Krone",
  "displayName-count-other": "Schwedische Kronen",
  "symbol": "SEK",
  "symbol-alt-narrow": "kr"
},
"SGD": {
  "displayName": "Singapur-Dollar",
  "displayName-count-one": "Singapur-Dollar",
  "displayName-count-other": "Singapur-Dollar",
  "symbol": "SGD",
  "symbol-alt-narrow": "$"
},
"SHP": {
  "displayName": "St. Helena-Pfund",
  "displayName-count-one": "St. Helena-Pfund",
  "displayName-count-other": "St. Helena-Pfund",
  "symbol": "SHP",
  "symbol-alt-narrow": "£"
},
"SIT": {
  "displayName": "Slowenischer Tolar",
  "displayName-count-one": "Slowenischer Tolar",
  "displayName-count-other": "Slowenische Tolar",
  "symbol": "SIT"
},
"SKK": {
  "displayName": "Slowakische Krone",
  "displayName-count-one": "Slowakische Kronen",
  "displayName-count-other": "Slowakische Kronen",
  "symbol": "SKK"
},

```

```

    "SLL": {
      "displayName": "Sierra-leonischer Leone",
      "displayName-count-one": "Sierra-leonischer Leone",
      "displayName-count-other": "Sierra-leonische Leones",
      "symbol": "SLL"
    },
    "SOS": {
      "displayName": "Somalia-Schilling",
      "displayName-count-one": "Somalia-Schilling",
      "displayName-count-other": "Somalia-Schilling",
      "symbol": "SOS"
    },
    "SRD": {
      "displayName": "Suriname-Dollar",
      "displayName-count-one": "Suriname-Dollar",
      "displayName-count-other": "Suriname-Dollar",
      "symbol": "SRD",
      "symbol-alt-narrow": "$"
    },
    "SRG": {
      "displayName": "Suriname Gulden",
      "displayName-count-one": "Suriname-Gulden",
      "displayName-count-other": "Suriname-Gulden",
      "symbol": "SRG"
    },
    "SSP": {
      "displayName": "Südsudanesisches Pfund",
      "displayName-count-one": "Südsudanesisches Pfund",
      "displayName-count-other": "Südsudanesische Pfund",
      "symbol": "SSP",
      "symbol-alt-narrow": "£"
    },
    "STD": {
      "displayName": "São-toméischer Dobra",
      "displayName-count-one": "São-toméischer Dobra",
      "displayName-count-other": "São-toméische Dobra",
      "symbol": "STD",
      "symbol-alt-narrow": "Db"
    },
    "SUR": {
      "displayName": "Sowjetischer Rubel",
      "displayName-count-one": "Sowjetische Rubel",
      "displayName-count-other": "Sowjetische Rubel",
      "symbol": "SUR"
    },
    "SVC": {
      "displayName": "El Salvador Colon",
      "displayName-count-one": "El Salvador-Colon",
      "displayName-count-other": "El Salvador-Colon",
      "symbol": "SVC"
    },
    "SYP": {
      "displayName": "Syrisches Pfund",
      "displayName-count-one": "Syrisches Pfund",
      "displayName-count-other": "Syrische Pfund",
      "symbol": "SYP",
      "symbol-alt-narrow": "SYP"
    }
  }

```

```

    },
    "SZL": {
      "displayName": "Swasiländischer Lilangeni",
      "displayName-count-one": "Swasiländischer Lilangeni",
      "displayName-count-other": "Swasiländische Emalangeni",
      "symbol": "SZL"
    },
    "THB": {
      "displayName": "Thailändischer Baht",
      "displayName-count-one": "Thailändischer Baht",
      "displayName-count-other": "Thailändische Baht",
      "symbol": "฿",
      "symbol-alt-narrow": "฿"
    },
    "TJR": {
      "displayName": "Tadschikistan Rubel",
      "displayName-count-one": "Tadschikistan-Rubel",
      "displayName-count-other": "Tadschikistan-Rubel",
      "symbol": "TJR"
    },
    "TJS": {
      "displayName": "Tadschikistan-Somoni",
      "displayName-count-one": "Tadschikistan-Somoni",
      "displayName-count-other": "Tadschikistan-Somoni",
      "symbol": "TJS"
    },
    "TMM": {
      "displayName": "Turkmenistan-Manat (1993-2009)",
      "displayName-count-one": "Turkmenistan-Manat (1993-2009)",
      "displayName-count-other": "Turkmenistan-Manat (1993-2009)",
      "symbol": "TMM"
    },
    "TMT": {
      "displayName": "Turkmenistan-Manat",
      "displayName-count-one": "Turkmenistan-Manat",
      "displayName-count-other": "Turkmenistan-Manat",
      "symbol": "TMT"
    },
    "TND": {
      "displayName": "Tunesischer Dinar",
      "displayName-count-one": "Tunesischer Dinar",
      "displayName-count-other": "Tunesische Dinar",
      "symbol": "TND"
    },
    "TOP": {
      "displayName": "Tongaischer Paʻanga",
      "displayName-count-one": "Tongaischer Paʻanga",
      "displayName-count-other": "Tongaische Paʻanga",
      "symbol": "TOP",
      "symbol-alt-narrow": "T$"
    },
    "TPE": {
      "displayName": "Timor-Escudo",
      "displayName-count-one": "Timor-Escudo",
      "displayName-count-other": "Timor-Escudo",
      "symbol": "TPE"
    },
  },

```

```

"TRL": {
  "displayName": "Türkische Lira (1922-2005)",
  "displayName-count-one": "Türkische Lira (1922-2005)",
  "displayName-count-other": "Türkische Lira (1922-2005)",
  "symbol": "TRL"
},
"TRY": {
  "displayName": "Türkische Lira",
  "displayName-count-one": "Türkische Lira",
  "displayName-count-other": "Türkische Lira",
  "symbol": "TRY",
  "symbol-alt-narrow": "₺",
  "symbol-alt-variant": "TL"
},
"TTD": {
  "displayName": "Trinidad und Tobago-Dollar",
  "displayName-count-one": "Trinidad und Tobago-Dollar",
  "displayName-count-other": "Trinidad und Tobago-Dollar",
  "symbol": "TTD",
  "symbol-alt-narrow": "$"
},
"TWD": {
  "displayName": "Neuer Taiwan-Dollar",
  "displayName-count-one": "Neuer Taiwan-Dollar",
  "displayName-count-other": "Neue Taiwan-Dollar",
  "symbol": "NT$",
  "symbol-alt-narrow": "NT$"
},
"TZS": {
  "displayName": "Tansania-Schilling",
  "displayName-count-one": "Tansania-Schilling",
  "displayName-count-other": "Tansania-Schilling",
  "symbol": "TZS"
},
"UAH": {
  "displayName": "Ukrainische Hrywnja",
  "displayName-count-one": "Ukrainische Hrywnja",
  "displayName-count-other": "Ukrainische Hrywen",
  "symbol": "UAH",
  "symbol-alt-narrow": "₴"
},
"UAK": {
  "displayName": "Ukrainischer Karbovanetz",
  "displayName-count-one": "Ukrainische Karbovanetz",
  "displayName-count-other": "Ukrainische Karbovanetz",
  "symbol": "UAK"
},
"UGS": {
  "displayName": "Uganda-Schilling (1966-1987)",
  "displayName-count-one": "Uganda-Schilling (1966-1987)",
  "displayName-count-other": "Uganda-Schilling (1966-1987)",
  "symbol": "UGS"
},
"UGX": {
  "displayName": "Uganda-Schilling",
  "displayName-count-one": "Uganda-Schilling",
  "displayName-count-other": "Uganda-Schilling",

```

```

        "symbol": "UGX"
    },
    "USD": {
        "displayName": "US-Dollar",
        "displayName-count-one": "US-Dollar",
        "displayName-count-other": "US-Dollar",
        "symbol": "$",
        "symbol-alt-narrow": "$"
    },
    "USN": {
        "displayName": "US Dollar (Nächster Tag)",
        "displayName-count-one": "US-Dollar (Nächster Tag)",
        "displayName-count-other": "US-Dollar (Nächster Tag)",
        "symbol": "USN"
    },
    "USS": {
        "displayName": "US Dollar (Gleicher Tag)",
        "displayName-count-one": "US-Dollar (Gleicher Tag)",
        "displayName-count-other": "US-Dollar (Gleicher Tag)",
        "symbol": "USS"
    },
    "UYI": {
        "displayName": "Uruguayischer Peso (Indexierte
Rechnungseinheiten)",
        "displayName-count-one": "Uruguayischer Peso (Indexierte
Rechnungseinheiten)",
        "displayName-count-other": "Uruguayische Pesos (Indexierte
Rechnungseinheiten)",
        "symbol": "UYI"
    },
    "UYP": {
        "displayName": "Uruguayischer Peso (1975-1993)",
        "displayName-count-one": "Uruguayischer Peso (1975-1993)",
        "displayName-count-other": "Uruguayische Pesos (1975-1993)",
        "symbol": "UYP"
    },
    "UYU": {
        "displayName": "Uruguayischer Peso",
        "displayName-count-one": "Uruguayischer Peso",
        "displayName-count-other": "Uruguayische Pesos",
        "symbol": "UYU",
        "symbol-alt-narrow": "$"
    },
    "UZS": {
        "displayName": "Usbekistan-Sum",
        "displayName-count-one": "Usbekistan-Sum",
        "displayName-count-other": "Usbekistan-Sum",
        "symbol": "UZS"
    },
    "VEB": {
        "displayName": "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-one": "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-other": "Venezolanische Bolívares (1871-
2008)",
        "symbol": "VEB"
    },
    "VEF": {

```

```

        "displayName": "Venezolanischer Bolívar",
        "displayName-count-one": "Venezolanischer Bolívar",
        "displayName-count-other": "Venezolanische Bolívares",
        "symbol": "VEF",
        "symbol-alt-narrow": "Bs"
    },
    "VND": {
        "displayName": "Vietnamesischer Dong",
        "displayName-count-one": "Vietnamesischer Dong",
        "displayName-count-other": "Vietnamesische Dong",
        "symbol": "₫",
        "symbol-alt-narrow": "₫"
    },
    "VNN": {
        "displayName": "Vietnamesischer Dong (1978-1985)",
        "displayName-count-one": "Vietnamesischer Dong (1978-1985)",
        "displayName-count-other": "Vietnamesische Dong (1978-1985)",
        "symbol": "VNN"
    },
    "VUV": {
        "displayName": "Vanuatu-Vatu",
        "displayName-count-one": "Vanuatu-Vatu",
        "displayName-count-other": "Vanuatu-Vatu",
        "symbol": "VUV"
    },
    "WST": {
        "displayName": "Samoanischer Tala",
        "displayName-count-one": "Samoanischer Tala",
        "displayName-count-other": "Samoanische Tala",
        "symbol": "WST"
    },
    "XAF": {
        "displayName": "CFA-Franc (BEAC)",
        "displayName-count-one": "CFA-Franc (BEAC)",
        "displayName-count-other": "CFA-Franc (BEAC)",
        "symbol": "FCFA"
    },
    "XAG": {
        "displayName": "Unze Silber",
        "displayName-count-one": "Unze Silber",
        "displayName-count-other": "Unzen Silber",
        "symbol": "XAG"
    },
    "XAU": {
        "displayName": "Unze Gold",
        "displayName-count-one": "Unze Gold",
        "displayName-count-other": "Unzen Gold",
        "symbol": "XAU"
    },
    "XBA": {
        "displayName": "Europäische Rechnungseinheit",
        "displayName-count-one": "Europäische Rechnungseinheiten",
        "displayName-count-other": "Europäische Rechnungseinheiten",
        "symbol": "XBA"
    },
    "XBB": {
        "displayName": "Europäische Währungseinheit (XBB)",

```

```

        "displayName-count-one": "Europäische Währungseinheiten (XBB)",
        "displayName-count-other": "Europäische Währungseinheiten
(XBB) ",
        "symbol": "XBB"
    },
    "XBC": {
        "displayName": "Europäische Rechnungseinheit (XBC)",
        "displayName-count-one": "Europäische Rechnungseinheiten (XBC)",
        "displayName-count-other": "Europäische Rechnungseinheiten
(XBC) ",
        "symbol": "XBC"
    },
    "XBD": {
        "displayName": "Europäische Rechnungseinheit (XBD)",
        "displayName-count-one": "Europäische Rechnungseinheiten (XBD)",
        "displayName-count-other": "Europäische Rechnungseinheiten
(XBD) ",
        "symbol": "XBD"
    },
    "XCD": {
        "displayName": "Ostkaribischer Dollar",
        "displayName-count-one": "Ostkaribischer Dollar",
        "displayName-count-other": "Ostkaribische Dollar",
        "symbol": "EC$",
        "symbol-alt-narrow": "$"
    },
    "XDR": {
        "displayName": "Sonderziehungsrechte",
        "displayName-count-one": "Sonderziehungsrechte",
        "displayName-count-other": "Sonderziehungsrechte",
        "symbol": "XDR"
    },
    "XEU": {
        "displayName": "Europäische Währungseinheit (XEU)",
        "displayName-count-one": "Europäische Währungseinheiten (XEU)",
        "displayName-count-other": "Europäische Währungseinheiten
(XEU) ",
        "symbol": "XEU"
    },
    "XFO": {
        "displayName": "Französischer Gold-Franc",
        "displayName-count-one": "Französische Gold-Franc",
        "displayName-count-other": "Französische Gold-Franc",
        "symbol": "XFO"
    },
    "XFU": {
        "displayName": "Französischer UIC-Franc",
        "displayName-count-one": "Französische UIC-Franc",
        "displayName-count-other": "Französische UIC-Franc",
        "symbol": "XFU"
    },
    "XOF": {
        "displayName": "CFA-Franc (BCEAO) ",
        "displayName-count-one": "CFA-Franc (BCEAO) ",
        "displayName-count-other": "CFA-Francs (BCEAO) ",
        "symbol": "CFA"
    },
    },

```



```

"XPD": {
  "displayName": "Unze Palladium",
  "displayName-count-one": "Unze Palladium",
  "displayName-count-other": "Unzen Palladium",
  "symbol": "XPD"
},
"XPF": {
  "displayName": "CFP-Franc",
  "displayName-count-one": "CFP-Franc",
  "displayName-count-other": "CFP-Franc",
  "symbol": "CFPF"
},
"XPT": {
  "displayName": "Unze Platin",
  "displayName-count-one": "Unze Platin",
  "displayName-count-other": "Unzen Platin",
  "symbol": "XPT"
},
"XRE": {
  "displayName": "RINET Funds",
  "displayName-count-one": "RINET Funds",
  "displayName-count-other": "RINET Funds",
  "symbol": "XRE"
},
"XSU": {
  "displayName": "SUCRE",
  "displayName-count-one": "SUCRE",
  "displayName-count-other": "SUCRE",
  "symbol": "XSU"
},
"XTS": {
  "displayName": "Testwährung",
  "displayName-count-one": "Testwährung",
  "displayName-count-other": "Testwährung",
  "symbol": "XTS"
},
"XUA": {
  "displayName": "Rechnungseinheit der AfEB",
  "displayName-count-one": "Rechnungseinheit der AfEB",
  "displayName-count-other": "Rechnungseinheiten der AfEB",
  "symbol": "XUA"
},
"XXX": {
  "displayName": "Unbekannte Währung",
  "displayName-count-one": "(unbekannte Währung)",
  "displayName-count-other": "(unbekannte Währung)",
  "symbol": "XXX"
},
"YDD": {
  "displayName": "Jemen-Dinar",
  "displayName-count-one": "Jemen-Dinar",
  "displayName-count-other": "Jemen-Dinar",
  "symbol": "YDD"
},
"YER": {
  "displayName": "Jemen-Rial",
  "displayName-count-one": "Jemen-Rial",

```

```

        "displayName-count-other": "Jemen-Rial",
        "symbol": "YER"
    },
    "YUD": {
        "displayName": "Jugoslawischer Dinar (1966-1990)",
        "displayName-count-one": "Jugoslawischer Dinar (1966-1990)",
        "displayName-count-other": "Jugoslawische Dinar (1966-1990)",
        "symbol": "YUD"
    },
    "YUM": {
        "displayName": "Jugoslawischer Neuer Dinar (1994-2002)",
        "displayName-count-one": "Jugoslawischer Neuer Dinar (1994-
2002)",
        "displayName-count-other": "Jugoslawische Neue Dinar (1994-
2002)",
        "symbol": "YUM"
    },
    "YUN": {
        "displayName": "Jugoslawischer Dinar (konvertibel)",
        "displayName-count-one": "Jugoslawische Dinar (konvertibel)",
        "displayName-count-other": "Jugoslawische Dinar (konvertibel)",
        "symbol": "YUN"
    },
    "YUR": {
        "displayName": "Jugoslawischer reformierter Dinar (1992-1993)",
        "displayName-count-one": "Jugoslawischer reformierter Dinar
(1992-1993)",
        "displayName-count-other": "Jugoslawische reformierte Dinar
(1992-1993)",
        "symbol": "YUR"
    },
    "ZAL": {
        "displayName": "Südafrikanischer Rand (Finanz)",
        "displayName-count-one": "Südafrikanischer Rand (Finanz)",
        "displayName-count-other": "Südafrikanischer Rand (Finanz)",
        "symbol": "ZAL"
    },
    "ZAR": {
        "displayName": "Südafrikanischer Rand",
        "displayName-count-one": "Südafrikanischer Rand",
        "displayName-count-other": "Südafrikanische Rand",
        "symbol": "ZAR",
        "symbol-alt-narrow": "R"
    },
    "ZMK": {
        "displayName": "Kwacha (1968-2012)",
        "displayName-count-one": "Kwacha (1968-2012)",
        "displayName-count-other": "Kwacha (1968-2012)",
        "symbol": "ZMK"
    },
    "ZMW": {
        "displayName": "Kwacha",
        "displayName-count-one": "Kwacha",
        "displayName-count-other": "Kwacha",
        "symbol": "ZMW",
        "symbol-alt-narrow": "K"
    },
    },

```

```
"ZRN": {
  "displayName": "Zaire-Neuer Zaïre (1993-1998)",
  "displayName-count-one": "Zaire-Neuer Zaïre (1993-1998)",
  "displayName-count-other": "Zaire-Neue Zaïre (1993-1998)",
  "symbol": "ZRN"
},
"ZRZ": {
  "displayName": "Zaire-Zaïre (1971-1993)",
  "displayName-count-one": "Zaire-Zaïre (1971-1993)",
  "displayName-count-other": "Zaire-Zaïre (1971-1993)",
  "symbol": "ZRZ"
},
"ZWD": {
  "displayName": "Simbabwe-Dollar (1980-2008)",
  "displayName-count-one": "Simbabwe-Dollar (1980-2008)",
  "displayName-count-other": "Simbabwe-Dollar (1980-2008)",
  "symbol": "ZWD"
},
"ZWL": {
  "displayName": "Simbabwe-Dollar (2009)",
  "displayName-count-one": "Simbabwe-Dollar (2009)",
  "displayName-count-other": "Simbabwe-Dollar (2009)",
  "symbol": "ZWL"
},
"ZWR": {
  "displayName": "Simbabwe-Dollar (2008)",
  "displayName-count-one": "Simbabwe-Dollar (2008)",
  "displayName-count-other": "Simbabwe-Dollar (2008)",
  "symbol": "ZWR"
}
}
```

CURRENCIES.JSX

```
{
    "main";
    {
        "de";
        {
            "identity";
            {
                "version";
                {
                    "_number";
                    "$Revision: 13259 $",
                    "_cldrVersion";
                    "31";
                }
                "language";
                "de";
            }
            "numbers";
        }
    }
}
```

```

    {
      "currencies";
      {
        "ADP";
        {
          "displayName";
          "Andorranische Pesete",
          "displayName-count-one";
          "Andorranische Pesete",
          "displayName-count-other";
          "Andorranische Peseten",
          "symbol";
          "ADP";
        }
        "AED";
        {
          "displayName";
          "VAE-Dirham",
          "displayName-count-one";
          "VAE-Dirham",
          "displayName-count-other";
          "VAE-Dirham",
          "symbol";
          "AED";
        }
        "AFA";
        {
          "displayName";
          "Afghanische Afghani (1927-2002)",
          "displayName-count-one";
          "Afghanische Afghani (1927-2002)",
          "displayName-count-other";
          "Afghanische Afghani (1927-2002)",
          "symbol";
          "AFA";
        }
        "AFN";
        {
          "displayName";
          "Afghanischer Afghani",
          "displayName-count-one";
          "Afghanischer Afghani",
          "displayName-count-other";
          "Afghanische Afghani",
          "symbol";
          "AFN";
        }
        "ALK";
        {
          "displayName";
          "Albanischer Lek (1946-1965)",
          "displayName-count-one";
          "Albanischer Lek (1946-1965)",
          "displayName-count-other";
          "Albanische Lek (1946-1965)";
        }
        "ALL";
      }
    }

```

```

    {
      "displayName";
      "Albanischer Lek",
      "displayName-count-one";
      "Albanischer Lek",
      "displayName-count-other";
      "Albanische Lek",
      "symbol";
      "ALL";
    }
    "AMD";
    {
      "displayName";
      "Armenischer Dram",
      "displayName-count-one";
      "Armenischer Dram",
      "displayName-count-other";
      "Armenische Dram",
      "symbol";
      "AMD";
    }
    "ANG";
    {
      "displayName";
      "Niederländische-Antillen-Gulden",
      "displayName-count-one";
      "Niederländische-Antillen-Gulden",
      "displayName-count-other";
      "Niederländische-Antillen-Gulden",
      "symbol";
      "ANG";
    }
    "AOA";
    {
      "displayName";
      "Angolanischer Kwanza",
      "displayName-count-one";
      "Angolanischer Kwanza",
      "displayName-count-other";
      "Angolanische Kwanza",
      "symbol";
      "AOA",
      "symbol-alt-narrow";
      "Kz";
    }
    "AOK";
    {
      "displayName";
      "Angolanischer Kwanza (1977-1990)",
      "displayName-count-one";
      "Angolanischer Kwanza (1977-1990)",
      "displayName-count-other";
      "Angolanische Kwanza (1977-1990)",
      "symbol";
      "AOK";
    }
    "AON";

```

```

    {
      "displayName";
      "Angolanischer Neuer Kwanza (1990-2000)",
      "displayName-count-one";
      "Angolanischer Neuer Kwanza (1990-2000)",
      "displayName-count-other";
      "Angolanische Neue Kwanza (1990-2000)",
      "symbol";
      "AON";
    }
    "AOR";
    {
      "displayName";
      "Angolanischer Kwanza Reajustado (1995-1999)",
      "displayName-count-one";
      "Angolanischer Kwanza Reajustado (1995-1999)",
      "displayName-count-other";
      "Angolanische Kwanza Reajustado (1995-1999)",
      "symbol";
      "AOR";
    }
    "ARA";
    {
      "displayName";
      "Argentinischer Austral",
      "displayName-count-one";
      "Argentinischer Austral",
      "displayName-count-other";
      "Argentinische Austral",
      "symbol";
      "ARA";
    }
    "ARL";
    {
      "displayName";
      "Argentinischer Peso Ley (1970-1983)",
      "displayName-count-one";
      "Argentinischer Peso Ley (1970-1983)",
      "displayName-count-other";
      "Argentinische Pesos Ley (1970-1983)",
      "symbol";
      "ARL";
    }
    "ARM";
    {
      "displayName";
      "Argentinischer Peso (1881-1970)",
      "displayName-count-one";
      "Argentinischer Peso (1881-1970)",
      "displayName-count-other";
      "Argentinische Pesos (1881-1970)",
      "symbol";
      "ARM";
    }
    "ARP";
    {
      "displayName";

```

```

        "Argentinischer Peso (1983-1985)",
        "displayName-count-one";
        "Argentinischer Peso (1983-1985)",
        "displayName-count-other";
        "Argentinische Peso (1983-1985)",
        "symbol";
        "ARP";
    }
    "ARS";
    {
        "displayName";
        "Argentinischer Peso",
        "displayName-count-one";
        "Argentinischer Peso",
        "displayName-count-other";
        "Argentinische Pesos",
        "symbol";
        "ARS",
        "symbol-alt-narrow";
        "$";
    }
    "ATS";
    {
        "displayName";
        "Österreichischer Schilling",
        "displayName-count-one";
        "Österreichischer Schilling",
        "displayName-count-other";
        "Österreichische Schilling",
        "symbol";
        "öS";
    }
    "AUD";
    {
        "displayName";
        "Australischer Dollar",
        "displayName-count-one";
        "Australischer Dollar",
        "displayName-count-other";
        "Australische Dollar",
        "symbol";
        "AU$",
        "symbol-alt-narrow";
        "$";
    }
    "AWG";
    {
        "displayName";
        "Aruba-Florin",
        "displayName-count-one";
        "Aruba-Florin",
        "displayName-count-other";
        "Aruba-Florin",
        "symbol";
        "AWG";
    }
    "AZM";

```

```

        {
            "displayName";
            "Aserbaidtschan-Manat (1993-2006)",
            "displayName-count-one";
            "Aserbaidtschan-Manat (1993-2006)",
            "displayName-count-other";
            "Aserbaidtschan-Manat (1993-2006)",
            "symbol";
            "AZM";
        }
        "AZN";
        {
            "displayName";
            "Aserbaidtschan-Manat",
            "displayName-count-one";
            "Aserbaidtschan-Manat",
            "displayName-count-other";
            "Aserbaidtschan-Manat",
            "symbol";
            "AZN";
        }
        "BAD";
        {
            "displayName";
            "Bosnien und Herzegowina Dinar (1992-1994)",
            "displayName-count-one";
            "Bosnien und Herzegowina Dinar (1992-1994)",
            "displayName-count-other";
            "Bosnien und Herzegowina Dinar (1992-1994)",
            "symbol";
            "BAD";
        }
        "BAM";
        {
            "displayName";
            "Bosnien und Herzegowina Konvertierbare Mark",
            "displayName-count-one";
            "Bosnien und Herzegowina Konvertierbare Mark",
            "displayName-count-other";
            "Bosnien und Herzegowina Konvertierbare Mark",
            "symbol";
            "BAM",
            "symbol-alt-narrow";
            "KM";
        }
        "BAN";
        {
            "displayName";
            "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
            "displayName-count-one";
            "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
            "displayName-count-other";
            "Bosnien und Herzegowina Neue Dinar (1994-1997)",
            "symbol";
            "BAN";
        }
        "BBD";
    }

```



```

{
  "displayName";
  "Barbados-Dollar",
    "displayName-count-one";
  "Barbados-Dollar",
    "displayName-count-other";
  "Barbados-Dollar",
    "symbol";
  "BBD",
    "symbol-alt-narrow";
  "$";
}
"BDT";
{
  "displayName";
  "Bangladesch-Taka",
    "displayName-count-one";
  "Bangladesch-Taka",
    "displayName-count-other";
  "Bangladesch-Taka",
    "symbol";
  "BDT",
    "symbol-alt-narrow";
  "৳";
}
"BEC";
{
  "displayName";
  "Belgischer Franc (konvertibel)",
    "displayName-count-one";
  "Belgischer Franc (konvertibel)",
    "displayName-count-other";
  "Belgische Franc (konvertibel)",
    "symbol";
  "BEC";
}
"BEF";
{
  "displayName";
  "Belgischer Franc",
    "displayName-count-one";
  "Belgischer Franc",
    "displayName-count-other";
  "Belgische Franc",
    "symbol";
  "BEF";
}
"BEL";
{
  "displayName";
  "Belgischer Finanz-Franc",
    "displayName-count-one";
  "Belgischer Finanz-Franc",
    "displayName-count-other";
  "Belgische Finanz-Franc",
    "symbol";
}

```

```

        "BEL";
    }
    "BGL";
    {
        "displayName";
        "Bulgarische Lew (1962-1999)",
        "displayName-count-one";
        "Bulgarische Lew (1962-1999)",
        "displayName-count-other";
        "Bulgarische Lew (1962-1999)",
        "symbol";
        "BGL";
    }
    "BGM";
    {
        "displayName";
        "Bulgarischer Lew (1952-1962)",
        "displayName-count-one";
        "Bulgarischer Lew (1952-1962)",
        "displayName-count-other";
        "Bulgarische Lew (1952-1962)",
        "symbol";
        "BGK";
    }
    "BGN";
    {
        "displayName";
        "Bulgarischer Lew",
        "displayName-count-one";
        "Bulgarischer Lew",
        "displayName-count-other";
        "Bulgarische Lew",
        "symbol";
        "BGN";
    }
    "BGO";
    {
        "displayName";
        "Bulgarischer Lew (1879-1952)",
        "displayName-count-one";
        "Bulgarischer Lew (1879-1952)",
        "displayName-count-other";
        "Bulgarische Lew (1879-1952)",
        "symbol";
        "BGJ";
    }
    "BHD";
    {
        "displayName";
        "Bahrain-Dinar",
        "displayName-count-one";
        "Bahrain-Dinar",
        "displayName-count-other";
        "Bahrain-Dinar",
        "symbol";
        "BHD";
    }
}

```

```

"BIF";
{
  "displayName";
  "Burundi-Franc",
    "displayName-count-one";
  "Burundi-Franc",
    "displayName-count-other";
  "Burundi-Francs",
    "symbol";
  "BIF";
}
"BMD";
{
  "displayName";
  "Bermuda-Dollar",
    "displayName-count-one";
  "Bermuda-Dollar",
    "displayName-count-other";
  "Bermuda-Dollar",
    "symbol";
  "BMD",
    "symbol-alt-narrow";
  "$";
}
"BND";
{
  "displayName";
  "Brunei-Dollar",
    "displayName-count-one";
  "Brunei-Dollar",
    "displayName-count-other";
  "Brunei-Dollar",
    "symbol";
  "BND",
    "symbol-alt-narrow";
  "$";
}
"BOB";
{
  "displayName";
  "Bolivanischer Boliviano",
    "displayName-count-one";
  "Bolivanischer Boliviano",
    "displayName-count-other";
  "Bolivianische Bolivianos",
    "symbol";
  "BOB",
    "symbol-alt-narrow";
  "Bs";
}
"BOL";
{
  "displayName";
  "Bolivianischer Boliviano (1863-1963)",
    "displayName-count-one";
  "Bolivianischer Boliviano (1863-1963)",
    "displayName-count-other";
}

```

```

        "Bolivianische Bolivianos (1863-1963)",
        "symbol";
        "BOL";
    }
    "BOP";
    {
        "displayName";
        "Bolivianischer Peso",
        "displayName-count-one";
        "Bolivianischer Peso",
        "displayName-count-other";
        "Bolivianische Peso",
        "symbol";
        "BOP";
    }
    "BOV";
    {
        "displayName";
        "Boliviansiche Mvdol",
        "displayName-count-one";
        "Boliviansiche Mvdol",
        "displayName-count-other";
        "Bolivianische Mvdol",
        "symbol";
        "BOV";
    }
    "BRB";
    {
        "displayName";
        "Brazilianischer Cruzeiro Novo (1967-1986)",
        "displayName-count-one";
        "Brazilianischer Cruzeiro Novo (1967-1986)",
        "displayName-count-other";
        "Brazilianische Cruzeiro Novo (1967-1986)",
        "symbol";
        "BRB";
    }
    "BRC";
    {
        "displayName";
        "Brazilianischer Cruzado (1986-1989)",
        "displayName-count-one";
        "Brazilianischer Cruzado (1986-1989)",
        "displayName-count-other";
        "Brazilianische Cruzado (1986-1989)",
        "symbol";
        "BRC";
    }
    "BRE";
    {
        "displayName";
        "Brazilianischer Cruzeiro (1990-1993)",
        "displayName-count-one";
        "Brazilianischer Cruzeiro (1990-1993)",
        "displayName-count-other";
        "Brazilianische Cruzeiro (1990-1993)",
        "symbol";
    }

```

```

        "BRE";
    }
    "BRL";
    {
        "displayName";
        "Brasilianischer Real",
        "displayName-count-one";
        "Brasilianischer Real",
        "displayName-count-other";
        "Brasilianische Real",
        "symbol";
        "R$";
        "symbol-alt-narrow";
        "R$";
    }
    "BRN";
    {
        "displayName";
        "Brasilianischer Cruzado Novo (1989-1990)",
        "displayName-count-one";
        "Brasilianischer Cruzado Novo (1989-1990)",
        "displayName-count-other";
        "Brasilianische Cruzado Novo (1989-1990)",
        "symbol";
        "BRN";
    }
    "BRR";
    {
        "displayName";
        "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-one";
        "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-other";
        "Brasilianische Cruzeiro (1993-1994)",
        "symbol";
        "BRR";
    }
    "BRZ";
    {
        "displayName";
        "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-one";
        "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-other";
        "Brasilianischer Cruzeiro (1942-1967)",
        "symbol";
        "BRZ";
    }
    "BSD";
    {
        "displayName";
        "Bahamas-Dollar",
        "displayName-count-one";
        "Bahamas-Dollar",
        "displayName-count-other";
        "Bahamas-Dollar",
        "symbol";
    }

```

```

        "BSD",
        "symbol-alt-narrow";
        "$";
    }
    "BTN";
    {
        "displayName";
        "Bhutan-Ngultrum",
        "displayName-count-one";
        "Bhutan-Ngultrum",
        "displayName-count-other";
        "Bhutan-Ngultrum",
        "symbol";
        "BTN";
    }
    "BUK";
    {
        "displayName";
        "Birmanischer Kyat",
        "displayName-count-one";
        "Birmanischer Kyat",
        "displayName-count-other";
        "Birmanische Kyat",
        "symbol";
        "BUK";
    }
    "BWP";
    {
        "displayName";
        "Botswanischer Pula",
        "displayName-count-one";
        "Botswanischer Pula",
        "displayName-count-other";
        "Botswanische Pula",
        "symbol";
        "BWP",
        "symbol-alt-narrow";
        "P";
    }
    "BYB";
    {
        "displayName";
        "Belarus-Rubel (1994-1999)",
        "displayName-count-one";
        "Belarus-Rubel (1994-1999)",
        "displayName-count-other";
        "Belarus-Rubel (1994-1999)",
        "symbol";
        "BYB";
    }
    "BYN";
    {
        "displayName";
        "Weißrussischer Rubel",
        "displayName-count-one";
        "Weißrussischer Rubel",
        "displayName-count-other";
    }

```

```

        "Weißrussische Rubel",
        "symbol";
        "BYN",
        "symbol-alt-narrow";
        "p.";
    }
    "BYR";
    {
        "displayName";
        "Weißrussischer Rubel (2000-2016)",
        "displayName-count-one";
        "Weißrussischer Rubel (2000-2016)",
        "displayName-count-other";
        "Weißrussische Rubel (2000-2016)",
        "symbol";
        "BYR";
    }
    "BZD";
    {
        "displayName";
        "Belize-Dollar",
        "displayName-count-one";
        "Belize-Dollar",
        "displayName-count-other";
        "Belize-Dollar",
        "symbol";
        "BZD",
        "symbol-alt-narrow";
        "$";
    }
    "CAD";
    {
        "displayName";
        "Kanadischer Dollar",
        "displayName-count-one";
        "Kanadischer Dollar",
        "displayName-count-other";
        "Kanadische Dollar",
        "symbol";
        "CA$",
        "symbol-alt-narrow";
        "$";
    }
    "CDF";
    {
        "displayName";
        "Kongo-Franc",
        "displayName-count-one";
        "Kongo-Franc",
        "displayName-count-other";
        "Kongo-Francs",
        "symbol";
        "CDF";
    }
    "CHE";
    {
        "displayName";

```

```

        "WIR-Euro",
        "displayName-count-one";
        "WIR-Euro",
        "displayName-count-other";
        "WIR-Euro",
        "symbol";
        "CHE";
    }
    "CHF";
    {
        "displayName";
        "Schweizer Franken",
        "displayName-count-one";
        "Schweizer Franken",
        "displayName-count-other";
        "Schweizer Franken",
        "symbol";
        "CHF";
    }
    "CHW";
    {
        "displayName";
        "WIR Franken",
        "displayName-count-one";
        "WIR Franken",
        "displayName-count-other";
        "WIR Franken",
        "symbol";
        "CHW";
    }
    "CLE";
    {
        "displayName";
        "Chilenischer Escudo",
        "displayName-count-one";
        "Chilenischer Escudo",
        "displayName-count-other";
        "Chilenische Escudo",
        "symbol";
        "CLE";
    }
    "CLF";
    {
        "displayName";
        "Chilenische Unidades de Fomento",
        "displayName-count-one";
        "Chilenische Unidades de Fomento",
        "displayName-count-other";
        "Chilenische Unidades de Fomento",
        "symbol";
        "CLF";
    }
    "CLP";
    {
        "displayName";
        "Chilenischer Peso",
        "displayName-count-one";

```



```

        "Chilenischer Peso",
        "displayName-count-other";
    "Chilenische Pesos",
        "symbol";
    "CLP",
        "symbol-alt-narrow";
    "$";
}
"CNX";
{
    "displayName";
    "Dollar der Chinesischen Volksbank",
        "displayName-count-one";
    "Dollar der Chinesischen Volksbank",
        "displayName-count-other";
    "Dollar der Chinesischen Volksbank",
        "symbol";
    "CNX";
}
"CNY";
{
    "displayName";
    "Renminbi Yuan",
        "displayName-count-one";
    "Chinesischer Yuan",
        "displayName-count-other";
    "Renminbi Yuan",
        "symbol";
    "CN¥",
        "symbol-alt-narrow";
    "¥";
}
"COP";
{
    "displayName";
    "Kolumbianischer Peso",
        "displayName-count-one";
    "Kolumbianischer Peso",
        "displayName-count-other";
    "Kolumbianische Pesos",
        "symbol";
    "COP",
        "symbol-alt-narrow";
    "$";
}
"COU";
{
    "displayName";
    "Kolumbianische Unidades de valor real",
        "displayName-count-one";
    "Kolumbianische Unidad de valor real",
        "displayName-count-other";
    "Kolumbianische Unidades de valor real",
        "symbol";
    "COU";
}
"CRC";

```

```

    {
      "displayName";
      "Costa-Rica-Colón",
        "displayName-count-one";
      "Costa-Rica-Colón",
        "displayName-count-other";
      "Costa-Rica-Colón",
        "symbol";
      "CRC",
        "symbol-alt-narrow";
      "₡";
    }
    "CSD";
    {
      "displayName";
      "Serbischer Dinar (2002-2006)",
        "displayName-count-one";
      "Serbischer Dinar (2002-2006)",
        "displayName-count-other";
      "Serbische Dinar (2002-2006)",
        "symbol";
      "CSD";
    }
    "CSK";
    {
      "displayName";
      "Tschechoslowakische Krone",
        "displayName-count-one";
      "Tschechoslowakische Kronen",
        "displayName-count-other";
      "Tschechoslowakische Kronen",
        "symbol";
      "CSK";
    }
    "CUC";
    {
      "displayName";
      "Kubanischer Peso (konvertibel)",
        "displayName-count-one";
      "Kubanischer Peso (konvertibel)",
        "displayName-count-other";
      "Kubanische Pesos (konvertibel)",
        "symbol";
      "CUC",
        "symbol-alt-narrow";
      "Cub$";
    }
    "CUP";
    {
      "displayName";
      "Kubanischer Peso",
        "displayName-count-one";
      "Kubanischer Peso",
        "displayName-count-other";
      "Kubanische Pesos",
        "symbol";
      "CUP",

```

```

        "symbol-alt-narrow";
        "$";
    }
    "CVE";
    {
        "displayName";
        "Cabo-Verde-Escudo",
        "displayName-count-one";
        "Cabo-Verde-Escudo",
        "displayName-count-other";
        "Cabo-Verde-Escudos",
        "symbol";
        "CVE";
    }
    "CYP";
    {
        "displayName";
        "Zypern-Pfund",
        "displayName-count-one";
        "Zypern Pfund",
        "displayName-count-other";
        "Zypern Pfund",
        "symbol";
        "CYP";
    }
    "CZK";
    {
        "displayName";
        "Tschechische Krone",
        "displayName-count-one";
        "Tschechische Krone",
        "displayName-count-other";
        "Tschechische Kronen",
        "symbol";
        "CZK",
        "symbol-alt-narrow";
        "Kč";
    }
    "DDM";
    {
        "displayName";
        "Mark der DDR",
        "displayName-count-one";
        "Mark der DDR",
        "displayName-count-other";
        "Mark der DDR",
        "symbol";
        "DDM";
    }
    "DEM";
    {
        "displayName";
        "Deutsche Mark",
        "displayName-count-one";
        "Deutsche Mark",
        "displayName-count-other";
        "Deutsche Mark",

```

```

        "symbol";
        "DM";
    }
    "DJF";
    {
        "displayName";
        "Dschibuti-Franc",
        "displayName-count-one";
        "Dschibuti-Franc",
        "displayName-count-other";
        "Dschibuti-Franc",
        "symbol";
        "DJF";
    }
    "DKK";
    {
        "displayName";
        "Dänische Krone",
        "displayName-count-one";
        "Dänische Krone",
        "displayName-count-other";
        "Dänische Kronen",
        "symbol";
        "DKK",
        "symbol-alt-narrow";
        "kr";
    }
    "DOP";
    {
        "displayName";
        "Dominikanischer Peso",
        "displayName-count-one";
        "Dominikanischer Peso",
        "displayName-count-other";
        "Dominikanische Pesos",
        "symbol";
        "DOP",
        "symbol-alt-narrow";
        "$";
    }
    "DZD";
    {
        "displayName";
        "Algerischer Dinar",
        "displayName-count-one";
        "Algerischer Dinar",
        "displayName-count-other";
        "Algerische Dinar",
        "symbol";
        "DZD";
    }
    "ECS";
    {
        "displayName";
        "Ecuadorianischer Sucre",
        "displayName-count-one";
        "Ecuadorianischer Sucre",

```

```

        "displayName-count-other";
        "Ecuadorianische Sucre",
        "symbol";
        "ECS";
    }
    "ECV";
    {
        "displayName";
        "Verrechnungseinheit für Ecuador",
        "displayName-count-one";
        "Verrechnungseinheiten für Ecuador",
        "displayName-count-other";
        "Verrechnungseinheiten für Ecuador",
        "symbol";
        "ECV";
    }
    "EEK";
    {
        "displayName";
        "Estnische Krone",
        "displayName-count-one";
        "Estnische Krone",
        "displayName-count-other";
        "Estnische Kronen",
        "symbol";
        "EEK";
    }
    "EGP";
    {
        "displayName";
        "Ägyptisches Pfund",
        "displayName-count-one";
        "Ägyptisches Pfund",
        "displayName-count-other";
        "Ägyptische Pfund",
        "symbol";
        "EGP",
        "symbol-alt-narrow";
        "£";
    }
    "ERN";
    {
        "displayName";
        "Eritreischer Nakfa",
        "displayName-count-one";
        "Eritreischer Nakfa",
        "displayName-count-other";
        "Eritreische Nakfa",
        "symbol";
        "ERN";
    }
    "ESA";
    {
        "displayName";
        "Spanische Peseta (A-Konten)",
        "displayName-count-one";
        "Spanische Peseta (A-Konten)",

```

```

        "displayName-count-other";
        "Spanische Peseten (A-Konten)",
        "symbol";
        "ESA";
    }
    "ESB";
    {
        "displayName";
        "Spanische Peseta (konvertibel)",
        "displayName-count-one";
        "Spanische Peseta (konvertibel)",
        "displayName-count-other";
        "Spanische Peseten (konvertibel)",
        "symbol";
        "ESB";
    }
    "ESP";
    {
        "displayName";
        "Spanische Peseta",
        "displayName-count-one";
        "Spanische Peseta",
        "displayName-count-other";
        "Spanische Peseten",
        "symbol";
        "ESP",
        "symbol-alt-narrow";
        "₧";
    }
    "ETB";
    {
        "displayName";
        "Äthiopischer Birr",
        "displayName-count-one";
        "Äthiopischer Birr",
        "displayName-count-other";
        "Äthiopische Birr",
        "symbol";
        "ETB";
    }
    "EUR";
    {
        "displayName";
        "Euro",
        "displayName-count-one";
        "Euro",
        "displayName-count-other";
        "Euro",
        "symbol";
        "€",
        "symbol-alt-narrow";
        "€";
    }
    "FIM";
    {
        "displayName";
        "Finnische Mark",

```

```

        "displayName-count-one";
        "Finnische Mark",
        "displayName-count-other";
        "Finnische Mark",
        "symbol";
        "FIM";
    }
    "FJD";
    {
        "displayName";
        "Fidschi-Dollar",
        "displayName-count-one";
        "Fidschi-Dollar",
        "displayName-count-other";
        "Fidschi-Dollar",
        "symbol";
        "FJD",
        "symbol-alt-narrow";
        "$";
    }
    "FKP";
    {
        "displayName";
        "Falkland-Pfund",
        "displayName-count-one";
        "Falkland-Pfund",
        "displayName-count-other";
        "Falkland-Pfund",
        "symbol";
        "FKP",
        "symbol-alt-narrow";
        "Fl£";
    }
    "FRF";
    {
        "displayName";
        "Französischer Franc",
        "displayName-count-one";
        "Französischer Franc",
        "displayName-count-other";
        "Französische Franc",
        "symbol";
        "FRF";
    }
    "GBP";
    {
        "displayName";
        "Britisches Pfund",
        "displayName-count-one";
        "Britisches Pfund",
        "displayName-count-other";
        "Britische Pfund",
        "symbol";
        "£",
        "symbol-alt-narrow";
        "£";
    }
}

```

```

    "GEK";
    {
        "displayName";
        "Georgischer Kupon Larit",
        "displayName-count-one";
        "Georgischer Kupon Larit",
        "displayName-count-other";
        "Georgische Kupon Larit",
        "symbol";
        "GEK";
    }
    "GEL";
    {
        "displayName";
        "Georgischer Lari",
        "displayName-count-one";
        "Georgischer Lari",
        "displayName-count-other";
        "Georgische Lari",
        "symbol";
        "GEL",
        "symbol-alt-narrow";
        "ლ",
        "symbol-alt-variant";
        "ლ";
    }
    "GHC";
    {
        "displayName";
        "Ghanaischer Cedi (1979-2007)",
        "displayName-count-one";
        "Ghanaischer Cedi (1979-2007)",
        "displayName-count-other";
        "Ghanaische Cedi (1979-2007)",
        "symbol";
        "GHC";
    }
    "GHS";
    {
        "displayName";
        "Ghanaischer Cedi",
        "displayName-count-one";
        "Ghanaischer Cedi",
        "displayName-count-other";
        "Ghanaische Cedi",
        "symbol";
        "GHS";
    }
    "GIP";
    {
        "displayName";
        "Gibraltar-Pfund",
        "displayName-count-one";
        "Gibraltar-Pfund",
        "displayName-count-other";
        "Gibraltar Pfund",
        "symbol";
    }

```



```

        "GIP",
        "symbol-alt-narrow";
        "£";
    }
    "GMD";
    {
        "displayName";
        "Gambia-Dalasi",
        "displayName-count-one";
        "Gambia-Dalasi",
        "displayName-count-other";
        "Gambia-Dalasi",
        "symbol";
        "GMD";
    }
    "GNF";
    {
        "displayName";
        "Guinea-Franc",
        "displayName-count-one";
        "Guinea-Franc",
        "displayName-count-other";
        "Guinea-Franc",
        "symbol";
        "GNF",
        "symbol-alt-narrow";
        "F.G.";
    }
    "GNS";
    {
        "displayName";
        "Guineischer Syli",
        "displayName-count-one";
        "Guineischer Syli",
        "displayName-count-other";
        "Guineische Syli",
        "symbol";
        "GNS";
    }
    "GQE";
    {
        "displayName";
        "Äquatorialguinea-Ekwele",
        "displayName-count-one";
        "Äquatorialguinea-Ekwele",
        "displayName-count-other";
        "Äquatorialguinea-Ekwele",
        "symbol";
        "GQE";
    }
    "GRD";
    {
        "displayName";
        "Griechische Drachme",
        "displayName-count-one";
        "Griechische Drachme",
        "displayName-count-other";
    }

```

```

        "Griechische Drachmen",
        "symbol";
        "GRD";
    }
    "GTQ";
    {
        "displayName";
        "Guatemaltekinscher Quetzal",
        "displayName-count-one";
        "Guatemaltekinscher Quetzal",
        "displayName-count-other";
        "Guatemaltekinsche Quetzales",
        "symbol";
        "GTQ",
        "symbol-alt-narrow";
        "Q";
    }
    "GWE";
    {
        "displayName";
        "Portugiesisch Guinea Escudo",
        "displayName-count-one";
        "Portugiesisch Guinea Escudo",
        "displayName-count-other";
        "Portugiesisch Guinea Escudo",
        "symbol";
        "GWE";
    }
    "GWP";
    {
        "displayName";
        "Guinea-Bissau Peso",
        "displayName-count-one";
        "Guinea-Bissau Peso",
        "displayName-count-other";
        "Guinea-Bissau Pesos",
        "symbol";
        "GWP";
    }
    "GYD";
    {
        "displayName";
        "Guyana-Dollar",
        "displayName-count-one";
        "Guyana-Dollar",
        "displayName-count-other";
        "Guyana-Dollar",
        "symbol";
        "GYD",
        "symbol-alt-narrow";
        "$";
    }
    "HKD";
    {
        "displayName";
        "Hongkong-Dollar",
        "displayName-count-one";

```

```

        "Hongkong-Dollar",
        "displayName-count-other";
    "Hongkong-Dollar",
        "symbol";
    "HK$",
        "symbol-alt-narrow";
    "$";
}
"HNL";
{
    "displayName";
    "Honduras-Lempira",
        "displayName-count-one";
    "Honduras-Lempira",
        "displayName-count-other";
    "Honduras-Lempira",
        "symbol";
    "HNL",
        "symbol-alt-narrow";
    "L";
}
"HRD";
{
    "displayName";
    "Kroatischer Dinar",
        "displayName-count-one";
    "Kroatischer Dinar",
        "displayName-count-other";
    "Kroatische Dinar",
        "symbol";
    "HRD";
}
"HRK";
{
    "displayName";
    "Kroatischer Kuna",
        "displayName-count-one";
    "Kroatischer Kuna",
        "displayName-count-other";
    "Kroatische Kuna",
        "symbol";
    "HRK",
        "symbol-alt-narrow";
    "kn";
}
"HTG";
{
    "displayName";
    "Haitianische Gourde",
        "displayName-count-one";
    "Haitianische Gourde",
        "displayName-count-other";
    "Haitianische Gourdes",
        "symbol";
    "HTG";
}
"HUF";

```

```

    {
      "displayName";
      "Ungarischer Forint",
        "displayName-count-one";
      "Ungarischer Forint",
        "displayName-count-other";
      "Ungarische Forint",
        "symbol";
      "HUF",
        "symbol-alt-narrow";
      "Ft";
    }
    "IDR";
    {
      "displayName";
      "Indonesische Rupiah",
        "displayName-count-one";
      "Indonesische Rupiah",
        "displayName-count-other";
      "Indonesische Rupiah",
        "symbol";
      "IDR",
        "symbol-alt-narrow";
      "Rp";
    }
    "IEP";
    {
      "displayName";
      "Irisches Pfund",
        "displayName-count-one";
      "Irisches Pfund",
        "displayName-count-other";
      "Irische Pfund",
        "symbol";
      "IEP";
    }
    "ILP";
    {
      "displayName";
      "Israelisches Pfund",
        "displayName-count-one";
      "Israelisches Pfund",
        "displayName-count-other";
      "Israelische Pfund",
        "symbol";
      "ILP";
    }
    "ILR";
    {
      "displayName";
      "Israelischer Schekel (1980-1985)",
        "displayName-count-one";
      "Israelischer Schekel (1980-1985)",
        "displayName-count-other";
      "Israelische Schekel (1980-1985)";
    }
    "ILS";

```

```

    {
      "displayName";
      "Israelischer Neuer Schekel",
      "displayName-count-one";
      "Israelischer Neuer Schekel",
      "displayName-count-other";
      "Israelische Neue Schekel",
      "symbol";
      "₪",
      "symbol-alt-narrow";
      "₪";
    }
    "INR";
    {
      "displayName";
      "Indische Rupie",
      "displayName-count-one";
      "Indische Rupie",
      "displayName-count-other";
      "Indische Rupien",
      "symbol";
      "₹",
      "symbol-alt-narrow";
      "₹";
    }
    "IQD";
    {
      "displayName";
      "Irakischer Dinar",
      "displayName-count-one";
      "Irakischer Dinar",
      "displayName-count-other";
      "Irakische Dinar",
      "symbol";
      "IQD";
    }
    "IRR";
    {
      "displayName";
      "Iranischer Rial",
      "displayName-count-one";
      "Iranischer Rial",
      "displayName-count-other";
      "Iranische Rial",
      "symbol";
      "IRR";
    }
    "ISJ";
    {
      "displayName";
      "Isländische Krone (1918-1981)",
      "displayName-count-one";
      "Isländische Krone (1918-1981)",
      "displayName-count-other";
      "Isländische Kronen (1918-1981)";
    }
    "ISK";

```

```

    {
      "displayName";
      "Isländische Krone",
        "displayName-count-one";
      "Isländische Krone",
        "displayName-count-other";
      "Isländische Kronen",
        "symbol";
      "ISK",
        "symbol-alt-narrow";
      "kr";
    }
    "ITL";
    {
      "displayName";
      "Italienische Lira",
        "displayName-count-one";
      "Italienische Lira",
        "displayName-count-other";
      "Italienische Lire",
        "symbol";
      "ITL";
    }
    "JMD";
    {
      "displayName";
      "Jamaika-Dollar",
        "displayName-count-one";
      "Jamaika-Dollar",
        "displayName-count-other";
      "Jamaika-Dollar",
        "symbol";
      "JMD",
        "symbol-alt-narrow";
      "$";
    }
    "JOD";
    {
      "displayName";
      "Jordanischer Dinar",
        "displayName-count-one";
      "Jordanischer Dinar",
        "displayName-count-other";
      "Jordanische Dinar",
        "symbol";
      "JOD";
    }
    "JPY";
    {
      "displayName";
      "Japanischer Yen",
        "displayName-count-one";
      "Japanischer Yen",
        "displayName-count-other";
      "Japanische Yen",
        "symbol";
      "¥";
    }
  
```

```

        "symbol-alt-narrow";
        "¥";
    }
    "KES";
    {
        "displayName";
        "Kenia-Schilling",
        "displayName-count-one";
        "Kenia-Schilling",
        "displayName-count-other";
        "Kenia-Schilling",
        "symbol";
        "KES";
    }
    "KGS";
    {
        "displayName";
        "Kirgisischer Som",
        "displayName-count-one";
        "Kirgisischer Som",
        "displayName-count-other";
        "Kirgisische Som",
        "symbol";
        "KGS";
    }
    "KHR";
    {
        "displayName";
        "Kambodschanischer Riel",
        "displayName-count-one";
        "Kambodschanischer Riel",
        "displayName-count-other";
        "Kambodschanische Riel",
        "symbol";
        "KHR",
        "symbol-alt-narrow";
        "៛";
    }
    "KMF";
    {
        "displayName";
        "Komoren-Franc",
        "displayName-count-one";
        "Komoren-Franc",
        "displayName-count-other";
        "Komoren-Francs",
        "symbol";
        "KMF",
        "symbol-alt-narrow";
        "FC";
    }
    "KPW";
    {
        "displayName";
        "Nordkoreanischer Won",
        "displayName-count-one";

```

```

        "Nordkoreanischer Won",
        "displayName-count-other";
    "Nordkoreanische Won",
        "symbol";
    "KPW",
        "symbol-alt-narrow";
    "₩";
}
"KRH";
{
    "displayName";
    "Südkoreanischer Hwan (1953-1962)",
        "displayName-count-one";
    "Südkoreanischer Hwan (1953-1962)",
        "displayName-count-other";
    "Südkoreanischer Hwan (1953-1962)",
        "symbol";
    "KRH";
}
"KRO";
{
    "displayName";
    "Südkoreanischer Won (1945-1953)",
        "displayName-count-one";
    "Südkoreanischer Won (1945-1953)",
        "displayName-count-other";
    "Südkoreanischer Won (1945-1953)",
        "symbol";
    "KRO";
}
"KRW";
{
    "displayName";
    "Südkoreanischer Won",
        "displayName-count-one";
    "Südkoreanischer Won",
        "displayName-count-other";
    "Südkoreanische Won",
        "symbol";
    "₩",
        "symbol-alt-narrow";
    "₩";
}
"KWD";
{
    "displayName";
    "Kuwait-Dinar",
        "displayName-count-one";
    "Kuwait-Dinar",
        "displayName-count-other";
    "Kuwait-Dinar",
        "symbol";
    "KWD";
}
"KYD";
{
    "displayName";

```



```

        "Kaiman-Dollar",
        "displayName-count-one";
        "Kaiman-Dollar",
        "displayName-count-other";
        "Kaiman-Dollar",
        "symbol";
        "KYD",
        "symbol-alt-narrow";
        "$";
    }
    "KZT";
    {
        "displayName";
        "Kasachischer Tenge",
        "displayName-count-one";
        "Kasachischer Tenge",
        "displayName-count-other";
        "Kasachische Tenge",
        "symbol";
        "KZT",
        "symbol-alt-narrow";
        "Т";
    }
    "LAK";
    {
        "displayName";
        "Laotischer Kip",
        "displayName-count-one";
        "Laotischer Kip",
        "displayName-count-other";
        "Laotische Kip",
        "symbol";
        "LAK",
        "symbol-alt-narrow";
        "₭";
    }
    "LBP";
    {
        "displayName";
        "Libanesisches Pfund",
        "displayName-count-one";
        "Libanesisches Pfund",
        "displayName-count-other";
        "Libanesische Pfund",
        "symbol";
        "LBP",
        "symbol-alt-narrow";
        "L£";
    }
    "LKR";
    {
        "displayName";
        "Sri-Lanka-Rupie",
        "displayName-count-one";
        "Sri-Lanka-Rupie",
        "displayName-count-other";
        "Sri-Lanka-Rupien",

```

```

        "symbol";
        "LKR",
        "symbol-alt-narrow";
        "Rs";
    }
    "LRD";
    {
        "displayName";
        "Liberianischer Dollar",
        "displayName-count-one";
        "Liberianischer Dollar",
        "displayName-count-other";
        "Liberianische Dollar",
        "symbol";
        "LRD",
        "symbol-alt-narrow";
        "$";
    }
    "LSL";
    {
        "displayName";
        "Loti",
        "displayName-count-one";
        "Loti",
        "displayName-count-other";
        "Loti",
        "symbol";
        "LSL";
    }
    "LTL";
    {
        "displayName";
        "Litauischer Litas",
        "displayName-count-one";
        "Litauischer Litas",
        "displayName-count-other";
        "Litauische Litas",
        "symbol";
        "LTL",
        "symbol-alt-narrow";
        "Lt";
    }
    "LTT";
    {
        "displayName";
        "Litauischer Talonas",
        "displayName-count-one";
        "Litauische Talonas",
        "displayName-count-other";
        "Litauische Talonas",
        "symbol";
        "LTT";
    }
    "LUC";
    {
        "displayName";
        "Luxemburgischer Franc (konvertibel)",

```

```

        "displayName-count-one";
        "Luxemburgische Franc (konvertibel)",
        "displayName-count-other";
        "Luxemburgische Franc (konvertibel)",
        "symbol";
        "LUC";
    }
    "LUF";
    {
        "displayName";
        "Luxemburgischer Franc",
        "displayName-count-one";
        "Luxemburgische Franc",
        "displayName-count-other";
        "Luxemburgische Franc",
        "symbol";
        "LUF";
    }
    "LUL";
    {
        "displayName";
        "Luxemburgischer Finanz-Franc",
        "displayName-count-one";
        "Luxemburgische Finanz-Franc",
        "displayName-count-other";
        "Luxemburgische Finanz-Franc",
        "symbol";
        "LUL";
    }
    "LVL";
    {
        "displayName";
        "Lettischer Lats",
        "displayName-count-one";
        "Lettischer Lats",
        "displayName-count-other";
        "Lettische Lats",
        "symbol";
        "LVL",
        "symbol-alt-narrow";
        "Ls";
    }
    "LVR";
    {
        "displayName";
        "Lettischer Rubel",
        "displayName-count-one";
        "Lettische Rubel",
        "displayName-count-other";
        "Lettische Rubel",
        "symbol";
        "LVR";
    }
    "LYD";
    {
        "displayName";
        "Libyscher Dinar",

```

```

        "displayName-count-one";
        "Libyscher Dinar",
        "displayName-count-other";
        "Libysche Dinar",
        "symbol";
        "LYD";
    }
    "MAD";
    {
        "displayName";
        "Marokkanischer Dirham",
        "displayName-count-one";
        "Marokkanischer Dirham",
        "displayName-count-other";
        "Marokkanische Dirham",
        "symbol";
        "MAD";
    }
    "MAF";
    {
        "displayName";
        "Marokkanischer Franc",
        "displayName-count-one";
        "Marokkanische Franc",
        "displayName-count-other";
        "Marokkanische Franc",
        "symbol";
        "MAF";
    }
    "MCF";
    {
        "displayName";
        "Monegassischer Franc",
        "displayName-count-one";
        "Monegassischer Franc",
        "displayName-count-other";
        "Monegassische Franc",
        "symbol";
        "MCF";
    }
    "MDC";
    {
        "displayName";
        "Moldau-Cupon",
        "displayName-count-one";
        "Moldau-Cupon",
        "displayName-count-other";
        "Moldau-Cupon",
        "symbol";
        "MDC";
    }
    "MDL";
    {
        "displayName";
        "Moldau-Leu",
        "displayName-count-one";
        "Moldau-Leu",

```

```

        "displayName-count-other";
        "Moldau-Leu",
        "symbol";
        "MDL";
    }
    "MGA";
    {
        "displayName";
        "Madagaskar-Ariary",
        "displayName-count-one";
        "Madagaskar-Ariary",
        "displayName-count-other";
        "Madagaskar-Ariary",
        "symbol";
        "MGA",
        "symbol-alt-narrow";
        "Ar";
    }
    "MGF";
    {
        "displayName";
        "Madagaskar-Franc",
        "displayName-count-one";
        "Madagaskar-Franc",
        "displayName-count-other";
        "Madagaskar-Franc",
        "symbol";
        "MGF";
    }
    "MKD";
    {
        "displayName";
        "Mazedonischer Denar",
        "displayName-count-one";
        "Mazedonischer Denar",
        "displayName-count-other";
        "Mazedonische Denari",
        "symbol";
        "MKD";
    }
    "MKN";
    {
        "displayName";
        "Mazedonischer Denar (1992-1993)",
        "displayName-count-one";
        "Mazedonischer Denar (1992-1993)",
        "displayName-count-other";
        "Mazedonische Denar (1992-1993)",
        "symbol";
        "MKN";
    }
    "MLF";
    {
        "displayName";
        "Malischer Franc",
        "displayName-count-one";
        "Malische Franc",

```

```

        "displayName-count-other";
        "Malische Franc",
        "symbol";
        "MLF";
    }
    "MMK";
    {
        "displayName";
        "Myanmarischer Kyat",
        "displayName-count-one";
        "Myanmarischer Kyat",
        "displayName-count-other";
        "Myanmarische Kyat",
        "symbol";
        "MMK",
        "symbol-alt-narrow";
        "K";
    }
    "MNT";
    {
        "displayName";
        "Mongolischer Tögrög",
        "displayName-count-one";
        "Mongolischer Tögrög",
        "displayName-count-other";
        "Mongolische Tögrög",
        "symbol";
        "MNT",
        "symbol-alt-narrow";
        "₮";
    }
    "MOP";
    {
        "displayName";
        "Macao-Pataca",
        "displayName-count-one";
        "Macao-Pataca",
        "displayName-count-other";
        "Macao-Pataca",
        "symbol";
        "MOP";
    }
    "MRO";
    {
        "displayName";
        "Mauretanischer Ouguiya",
        "displayName-count-one";
        "Mauretanischer Ouguiya",
        "displayName-count-other";
        "Mauretanische Ouguiya",
        "symbol";
        "MRO";
    }
    "MTL";
    {
        "displayName";
        "Maltesische Lira",

```

```

        "displayName-count-one";
        "Maltesische Lira",
        "displayName-count-other";
        "Maltesische Lira",
        "symbol";
        "MTL";
    }
    "MTP";
    {
        "displayName";
        "Maltesisches Pfund",
        "displayName-count-one";
        "Maltesische Pfund",
        "displayName-count-other";
        "Maltesische Pfund",
        "symbol";
        "MTP";
    }
    "MUR";
    {
        "displayName";
        "Mauritius-Rupie",
        "displayName-count-one";
        "Mauritius-Rupie",
        "displayName-count-other";
        "Mauritius-Rupien",
        "symbol";
        "MUR",
        "symbol-alt-narrow";
        "Rs";
    }
    "MVP";
    {
        "displayName";
        "Malediven-Rupie (alt)",
        "displayName-count-one";
        "Malediven-Rupie (alt)",
        "displayName-count-other";
        "Malediven-Rupien (alt)";
    }
    "MVR";
    {
        "displayName";
        "Malediven-Rufiyaa",
        "displayName-count-one";
        "Malediven-Rufiyaa",
        "displayName-count-other";
        "Malediven-Rupien",
        "symbol";
        "MVR";
    }
    "MWK";
    {
        "displayName";
        "Malawi-Kwacha",
        "displayName-count-one";
        "Malawi-Kwacha",

```

```

        "displayName-count-other";
        "Malawi-Kwacha",
        "symbol";
        "MWK";
    }
    "MXN";
    {
        "displayName";
        "Mexikanischer Peso",
        "displayName-count-one";
        "Mexikanischer Peso",
        "displayName-count-other";
        "Mexikanische Pesos",
        "symbol";
        "MX$";
        "symbol-alt-narrow";
        "$";
    }
    "MXP";
    {
        "displayName";
        "Mexikanischer Silber-Peso (1861-1992)",
        "displayName-count-one";
        "Mexikanische Silber-Peso (1861-1992)",
        "displayName-count-other";
        "Mexikanische Silber-Pesos (1861-1992)",
        "symbol";
        "MXP";
    }
    "MXV";
    {
        "displayName";
        "Mexicanischer Unidad de Inversion (UDI)",
        "displayName-count-one";
        "Mexicanischer Unidad de Inversion (UDI)",
        "displayName-count-other";
        "Mexikanische Unidad de Inversion (UDI)",
        "symbol";
        "MXV";
    }
    "MYR";
    {
        "displayName";
        "Malaysischer Ringgit",
        "displayName-count-one";
        "Malaysischer Ringgit",
        "displayName-count-other";
        "Malaysische Ringgit",
        "symbol";
        "MYR";
        "symbol-alt-narrow";
        "RM";
    }
    "MZE";
    {
        "displayName";
        "Mosambikanischer Escudo",

```



```

        "displayName-count-one";
        "Mozambikanische Escudo",
        "displayName-count-other";
        "Mozambikanische Escudo",
        "symbol";
        "MZE";
    }
    "MZM";
    {
        "displayName";
        "Mosambikanischer Metical (1980-2006)",
        "displayName-count-one";
        "Mosambikanischer Metical (1980-2006)",
        "displayName-count-other";
        "Mosambikanische Meticaïs (1980-2006)",
        "symbol";
        "MZM";
    }
    "MZN";
    {
        "displayName";
        "Mosambikanischer Metical",
        "displayName-count-one";
        "Mosambikanischer Metical",
        "displayName-count-other";
        "Mosambikanische Meticaïs",
        "symbol";
        "MZN";
    }
    "NAD";
    {
        "displayName";
        "Namibia-Dollar",
        "displayName-count-one";
        "Namibia-Dollar",
        "displayName-count-other";
        "Namibia-Dollar",
        "symbol";
        "NAD",
        "symbol-alt-narrow";
        "$";
    }
    "NGN";
    {
        "displayName";
        "Nigerianischer Naira",
        "displayName-count-one";
        "Nigerianischer Naira",
        "displayName-count-other";
        "Nigerianische Naira",
        "symbol";
        "NGN",
        "symbol-alt-narrow";
        "₦";
    }
    "NIC";
    {

```

```

        "displayName";
        "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-one";
        "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-other";
        "Nicaraguanische Córdoba (1988-1991)",
        "symbol";
        "NIC";
    }
    "NIO";
    {
        "displayName";
        "Nicaragua-Córdoba",
        "displayName-count-one";
        "Nicaragua-Córdoba",
        "displayName-count-other";
        "Nicaragua-Córdobas",
        "symbol";
        "NIO",
        "symbol-alt-narrow";
        "C$";
    }
    "NLG";
    {
        "displayName";
        "Niederländischer Gulden",
        "displayName-count-one";
        "Niederländischer Gulden",
        "displayName-count-other";
        "Niederländische Gulden",
        "symbol";
        "NLG";
    }
    "NOK";
    {
        "displayName";
        "Norwegische Krone",
        "displayName-count-one";
        "Norwegische Krone",
        "displayName-count-other";
        "Norwegische Kronen",
        "symbol";
        "NOK",
        "symbol-alt-narrow";
        "kr";
    }
    "NPR";
    {
        "displayName";
        "Nepalesische Rupie",
        "displayName-count-one";
        "Nepalesische Rupie",
        "displayName-count-other";
        "Nepalesische Rupien",
        "symbol";
        "NPR",
        "symbol-alt-narrow";
    }

```

```

        "Rs";
    }
    "NZD";
    {
        "displayName";
        "Neuseeland-Dollar",
        "displayName-count-one";
        "Neuseeland-Dollar",
        "displayName-count-other";
        "Neuseeland-Dollar",
        "symbol";
        "NZ$",
        "symbol-alt-narrow";
        "$";
    }
    "OMR";
    {
        "displayName";
        "Omanischer Rial",
        "displayName-count-one";
        "Omanischer Rial",
        "displayName-count-other";
        "Omanische Rials",
        "symbol";
        "OMR";
    }
    "PAB";
    {
        "displayName";
        "Panamaischer Balboa",
        "displayName-count-one";
        "Panamaischer Balboa",
        "displayName-count-other";
        "Panamaische Balboas",
        "symbol";
        "PAB";
    }
    "PEI";
    {
        "displayName";
        "Peruanischer Inti",
        "displayName-count-one";
        "Peruanische Inti",
        "displayName-count-other";
        "Peruanische Inti",
        "symbol";
        "PEI";
    }
    "PEN";
    {
        "displayName";
        "Peruanischer Sol",
        "displayName-count-one";
        "Peruanischer Sol",
        "displayName-count-other";
        "Peruanische Sol",
        "symbol";
    }

```

```

        "PEN";
    }
    "PES";
    {
        "displayName";
        "Peruanischer Sol (1863-1965)",
        "displayName-count-one";
        "Peruanischer Sol (1863-1965)",
        "displayName-count-other";
        "Peruanische Sol (1863-1965)",
        "symbol";
        "PES";
    }
    "PGK";
    {
        "displayName";
        "Papua-Neuguineischer Kina",
        "displayName-count-one";
        "Papua-Neuguineischer Kina",
        "displayName-count-other";
        "Papua-Neuguineische Kina",
        "symbol";
        "PGK";
    }
    "PHP";
    {
        "displayName";
        "Philippinischer Peso",
        "displayName-count-one";
        "Philippinischer Peso",
        "displayName-count-other";
        "Philippinische Pesos",
        "symbol";
        "PHP",
        "symbol-alt-narrow";
        "₱";
    }
    "PKR";
    {
        "displayName";
        "Pakistanische Rupie",
        "displayName-count-one";
        "Pakistanische Rupie",
        "displayName-count-other";
        "Pakistanische Rupien",
        "symbol";
        "PKR",
        "symbol-alt-narrow";
        "Rs";
    }
    "PLN";
    {
        "displayName";
        "Polnischer Złoty",
        "displayName-count-one";
        "Polnischer Złoty",
        "displayName-count-other";
    }

```

```

        "Polnische Złoty",
        "symbol";
        "PLN",
        "symbol-alt-narrow";
        "zł";
    }
    "PLZ";
    {
        "displayName";
        "Polnischer Zloty (1950-1995)",
        "displayName-count-one";
        "Polnischer Zloty (1950-1995)",
        "displayName-count-other";
        "Polnische Zloty (1950-1995)",
        "symbol";
        "PLZ";
    }
    "PTE";
    {
        "displayName";
        "Portugiesischer Escudo",
        "displayName-count-one";
        "Portugiesische Escudo",
        "displayName-count-other";
        "Portugiesische Escudo",
        "symbol";
        "PTE";
    }
    "PYG";
    {
        "displayName";
        "Paraguayischer Guaraní",
        "displayName-count-one";
        "Paraguayischer Guaraní",
        "displayName-count-other";
        "Paraguayische Guaraníes",
        "symbol";
        "PYG",
        "symbol-alt-narrow";
        "₲";
    }
    "QAR";
    {
        "displayName";
        "Katar-Riyal",
        "displayName-count-one";
        "Katar-Riyal",
        "displayName-count-other";
        "Katar-Riyal",
        "symbol";
        "QAR";
    }
    "RHD";
    {
        "displayName";
        "Rhodesischer Dollar",
        "displayName-count-one";

```

```

        "Rhodesische Dollar",
        "displayName-count-other";
        "Rhodesische Dollar",
        "symbol";
        "RHD";
    }
    "ROL";
    {
        "displayName";
        "Rumänischer Leu (1952-2006)",
        "displayName-count-one";
        "Rumänischer Leu (1952-2006)",
        "displayName-count-other";
        "Rumänische Leu (1952-2006)",
        "symbol";
        "ROL";
    }
    "RON";
    {
        "displayName";
        "Rumänischer Leu",
        "displayName-count-one";
        "Rumänischer Leu",
        "displayName-count-other";
        "Rumänische Leu",
        "symbol";
        "RON",
        "symbol-alt-narrow";
        "L";
    }
    "RSD";
    {
        "displayName";
        "Serbischer Dinar",
        "displayName-count-one";
        "Serbischer Dinar",
        "displayName-count-other";
        "Serbische Dinaren",
        "symbol";
        "RSD";
    }
    "RUB";
    {
        "displayName";
        "Russischer Rubel",
        "displayName-count-one";
        "Russischer Rubel",
        "displayName-count-other";
        "Russische Rubel",
        "symbol";
        "RUB",
        "symbol-alt-narrow";
        "₽";
    }
    "RUR";
    {
        "displayName";

```

```

        "Russischer Rubel (1991-1998)",
        "displayName-count-one";
        "Russischer Rubel (1991-1998)",
        "displayName-count-other";
        "Russische Rubel (1991-1998)",
        "symbol";
        "RUR",
        "symbol-alt-narrow";
        "p.";
    }
    "RWF";
    {
        "displayName";
        "Ruanda-Franc",
        "displayName-count-one";
        "Ruanda-Franc",
        "displayName-count-other";
        "Ruanda-Francis",
        "symbol";
        "RWF",
        "symbol-alt-narrow";
        "F.Rw";
    }
    "SAR";
    {
        "displayName";
        "Saudi-Rial",
        "displayName-count-one";
        "Saudi-Rial",
        "displayName-count-other";
        "Saudi-Rial",
        "symbol";
        "SAR";
    }
    "SBD";
    {
        "displayName";
        "Salomonen-Dollar",
        "displayName-count-one";
        "Salomonen-Dollar",
        "displayName-count-other";
        "Salomonen-Dollar",
        "symbol";
        "SBD",
        "symbol-alt-narrow";
        "$";
    }
    "SCR";
    {
        "displayName";
        "Seychellen-Rupie",
        "displayName-count-one";
        "Seychellen-Rupie",
        "displayName-count-other";
        "Seychellen-Rupien",
        "symbol";
        "SCR";
    }

```

```

    }
    "SDD";
    {
        "displayName";
        "Sudanesischer Dinar (1992-2007)",
        "displayName-count-one";
        "Sudanesischer Dinar (1992-2007)",
        "displayName-count-other";
        "Sudanesische Dinar (1992-2007)",
        "symbol";
        "SDD";
    }
    "SDG";
    {
        "displayName";
        "Sudanesisches Pfund",
        "displayName-count-one";
        "Sudanesisches Pfund",
        "displayName-count-other";
        "Sudanesische Pfund",
        "symbol";
        "SDG";
    }
    "SDP";
    {
        "displayName";
        "Sudanesisches Pfund (1957-1998)",
        "displayName-count-one";
        "Sudanesisches Pfund (1957-1998)",
        "displayName-count-other";
        "Sudanesische Pfund (1957-1998)",
        "symbol";
        "SDP";
    }
    "SEK";
    {
        "displayName";
        "Schwedische Krone",
        "displayName-count-one";
        "Schwedische Krone",
        "displayName-count-other";
        "Schwedische Kronen",
        "symbol";
        "SEK",
        "symbol-alt-narrow";
        "kr";
    }
    "SGD";
    {
        "displayName";
        "Singapur-Dollar",
        "displayName-count-one";
        "Singapur-Dollar",
        "displayName-count-other";
        "Singapur-Dollar",
        "symbol";
        "SGD",
    }

```



```

        "symbol-alt-narrow";
        "$";
    }
    "SHP";
    {
        "displayName";
        "St. Helena-Pfund",
        "displayName-count-one";
        "St. Helena-Pfund",
        "displayName-count-other";
        "St. Helena-Pfund",
        "symbol";
        "SHP",
        "symbol-alt-narrow";
        "£";
    }
    "SIT";
    {
        "displayName";
        "Slowenischer Tolar",
        "displayName-count-one";
        "Slowenischer Tolar",
        "displayName-count-other";
        "Slowenische Tolar",
        "symbol";
        "SIT";
    }
    "SKK";
    {
        "displayName";
        "Slowakische Krone",
        "displayName-count-one";
        "Slowakische Kronen",
        "displayName-count-other";
        "Slowakische Kronen",
        "symbol";
        "SKK";
    }
    "SLL";
    {
        "displayName";
        "Sierra-leonischer Leone",
        "displayName-count-one";
        "Sierra-leonischer Leone",
        "displayName-count-other";
        "Sierra-leonische Leones",
        "symbol";
        "SLL";
    }
    "SOS";
    {
        "displayName";
        "Somalia-Schilling",
        "displayName-count-one";
        "Somalia-Schilling",
        "displayName-count-other";
        "Somalia-Schilling",

```

```

        "symbol";
        "SOS";
    }
    "SRD";
    {
        "displayName";
        "Suriname-Dollar",
        "displayName-count-one";
        "Suriname-Dollar",
        "displayName-count-other";
        "Suriname-Dollar",
        "symbol";
        "SRD",
        "symbol-alt-narrow";
        "$";
    }
    "SRG";
    {
        "displayName";
        "Suriname Gulden",
        "displayName-count-one";
        "Suriname-Gulden",
        "displayName-count-other";
        "Suriname-Gulden",
        "symbol";
        "SRG";
    }
    "SSP";
    {
        "displayName";
        "Südsudanesisches Pfund",
        "displayName-count-one";
        "Südsudanesisches Pfund",
        "displayName-count-other";
        "Südsudanesische Pfund",
        "symbol";
        "SSP",
        "symbol-alt-narrow";
        "£";
    }
    "STD";
    {
        "displayName";
        "São-toméischer Dobra",
        "displayName-count-one";
        "São-toméischer Dobra",
        "displayName-count-other";
        "São-toméische Dobra",
        "symbol";
        "STD",
        "symbol-alt-narrow";
        "Db";
    }
    "SUR";
    {
        "displayName";
        "Sowjetischer Rubel",

```

```

        "displayName-count-one";
        "Sowjetische Rubel",
        "displayName-count-other";
        "Sowjetische Rubel",
        "symbol";
        "SUR";
    }
    "SVC";
    {
        "displayName";
        "El Salvador Colon",
        "displayName-count-one";
        "El Salvador-Colon",
        "displayName-count-other";
        "El Salvador-Colon",
        "symbol";
        "SVC";
    }
    "SYP";
    {
        "displayName";
        "Syrisches Pfund",
        "displayName-count-one";
        "Syrisches Pfund",
        "displayName-count-other";
        "Syrische Pfund",
        "symbol";
        "SYP",
        "symbol-alt-narrow";
        "SYP";
    }
    "SZL";
    {
        "displayName";
        "Swasiländischer Lilangeni",
        "displayName-count-one";
        "Swasiländischer Lilangeni",
        "displayName-count-other";
        "Swasiländische Emalangeni",
        "symbol";
        "SZL";
    }
    "THB";
    {
        "displayName";
        "Thailändischer Baht",
        "displayName-count-one";
        "Thailändischer Baht",
        "displayName-count-other";
        "Thailändische Baht",
        "symbol";
        "฿",
        "symbol-alt-narrow";
        "฿";
    }
    "TJR";
    {

```

```

        "displayName";
        "Tadschikistan Rubel",
        "displayName-count-one";
        "Tadschikistan-Rubel",
        "displayName-count-other";
        "Tadschikistan-Rubel",
        "symbol";
        "TJR";
    }
    "TJS";
    {
        "displayName";
        "Tadschikistan-Somoni",
        "displayName-count-one";
        "Tadschikistan-Somoni",
        "displayName-count-other";
        "Tadschikistan-Somoni",
        "symbol";
        "TJS";
    }
    "TMM";
    {
        "displayName";
        "Turkmenistan-Manat (1993-2009)",
        "displayName-count-one";
        "Turkmenistan-Manat (1993-2009)",
        "displayName-count-other";
        "Turkmenistan-Manat (1993-2009)",
        "symbol";
        "TMM";
    }
    "TMT";
    {
        "displayName";
        "Turkmenistan-Manat",
        "displayName-count-one";
        "Turkmenistan-Manat",
        "displayName-count-other";
        "Turkmenistan-Manat",
        "symbol";
        "TMT";
    }
    "TND";
    {
        "displayName";
        "Tunesischer Dinar",
        "displayName-count-one";
        "Tunesischer Dinar",
        "displayName-count-other";
        "Tunesische Dinar",
        "symbol";
        "TND";
    }
    "TOP";
    {
        "displayName";
        "Tongaischer Pa'anga",

```

```

        "displayName-count-one";
        "Tongaischer Pa'anga",
        "displayName-count-other";
        "Tongaische Pa'anga",
        "symbol";
        "TOP",
        "symbol-alt-narrow";
        "T$";
    }
    "TPE";
    {
        "displayName";
        "Timor-Escudo",
        "displayName-count-one";
        "Timor-Escudo",
        "displayName-count-other";
        "Timor-Escudo",
        "symbol";
        "TPE";
    }
    "TRL";
    {
        "displayName";
        "Türkische Lira (1922-2005)",
        "displayName-count-one";
        "Türkische Lira (1922-2005)",
        "displayName-count-other";
        "Türkische Lira (1922-2005)",
        "symbol";
        "TRL";
    }
    "TRY";
    {
        "displayName";
        "Türkische Lira",
        "displayName-count-one";
        "Türkische Lira",
        "displayName-count-other";
        "Türkische Lira",
        "symbol";
        "TRY",
        "symbol-alt-narrow";
        "₺",
        "symbol-alt-variant";
        "TL";
    }
    "TTD";
    {
        "displayName";
        "Trinidad und Tobago-Dollar",
        "displayName-count-one";
        "Trinidad und Tobago-Dollar",
        "displayName-count-other";
        "Trinidad und Tobago-Dollar",
        "symbol";
        "TTD",
        "symbol-alt-narrow";
    }

```

```

        "$";
    }
    "TWD";
    {
        "displayName";
        "Neuer Taiwan-Dollar",
        "displayName-count-one";
        "Neuer Taiwan-Dollar",
        "displayName-count-other";
        "Neue Taiwan-Dollar",
        "symbol";
        "NT$";
        "symbol-alt-narrow";
        "NT$";
    }
    "TZS";
    {
        "displayName";
        "Tansania-Schilling",
        "displayName-count-one";
        "Tansania-Schilling",
        "displayName-count-other";
        "Tansania-Schilling",
        "symbol";
        "TZS";
    }
    "UAH";
    {
        "displayName";
        "Ukrainische Hrywnja",
        "displayName-count-one";
        "Ukrainische Hrywnja",
        "displayName-count-other";
        "Ukrainische Hrywen",
        "symbol";
        "UAH";
        "symbol-alt-narrow";
        "₴";
    }
    "UAK";
    {
        "displayName";
        "Ukrainischer Karbovanetz",
        "displayName-count-one";
        "Ukrainische Karbovanetz",
        "displayName-count-other";
        "Ukrainische Karbovanetz",
        "symbol";
        "UAK";
    }
    "UGS";
    {
        "displayName";
        "Uganda-Schilling (1966-1987)",
        "displayName-count-one";
        "Uganda-Schilling (1966-1987)",
        "displayName-count-other";
    }

```

```

        "Uganda-Schilling (1966-1987)",
        "symbol";
        "UGS";
    }
    "UGX";
    {
        "displayName";
        "Uganda-Schilling",
        "displayName-count-one";
        "Uganda-Schilling",
        "displayName-count-other";
        "Uganda-Schilling",
        "symbol";
        "UGX";
    }
    "USD";
    {
        "displayName";
        "US-Dollar",
        "displayName-count-one";
        "US-Dollar",
        "displayName-count-other";
        "US-Dollar",
        "symbol";
        "$",
        "symbol-alt-narrow";
        "$";
    }
    "USN";
    {
        "displayName";
        "US Dollar (Nächster Tag)",
        "displayName-count-one";
        "US-Dollar (Nächster Tag)",
        "displayName-count-other";
        "US-Dollar (Nächster Tag)",
        "symbol";
        "USN";
    }
    "USS";
    {
        "displayName";
        "US Dollar (Gleicher Tag)",
        "displayName-count-one";
        "US-Dollar (Gleicher Tag)",
        "displayName-count-other";
        "US-Dollar (Gleicher Tag)",
        "symbol";
        "USS";
    }
    "UYI";
    {
        "displayName";
        "Uruguayischer Peso (Indexierte
Rechnungseinheiten)",
        "displayName-count-one";

```

```

    "Uruguayischer Peso (Indexierte
Rechnungseinheiten)",
        "displayName-count-other";
    "Uruguayische Pesos (Indexierte
Rechnungseinheiten)",
        "symbol";
    "UYI";
}
"UYP";
{
    "displayName";
    "Uruguayischer Peso (1975-1993)",
        "displayName-count-one";
    "Uruguayischer Peso (1975-1993)",
        "displayName-count-other";
    "Uruguayische Pesos (1975-1993)",
        "symbol";
    "UYP";
}
"UYU";
{
    "displayName";
    "Uruguayischer Peso",
        "displayName-count-one";
    "Uruguayischer Peso",
        "displayName-count-other";
    "Uruguayische Pesos",
        "symbol";
    "UYU",
        "symbol-alt-narrow";
    "$";
}
"UZS";
{
    "displayName";
    "Usbekistan-Sum",
        "displayName-count-one";
    "Usbekistan-Sum",
        "displayName-count-other";
    "Usbekistan-Sum",
        "symbol";
    "UZS";
}
"VEB";
{
    "displayName";
    "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-one";
    "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-other";
    "Venezolanische Bolívares (1871-2008)",
        "symbol";
    "VEB";
}
"VEF";
{
    "displayName";

```



```

        "Venezolanischer Bolívar",
        "displayName-count-one";
        "Venezolanischer Bolívar",
        "displayName-count-other";
        "Venezolanische Bolívares",
        "symbol";
        "VEF",
        "symbol-alt-narrow";
        "Bs";
    }
    "VND";
    {
        "displayName";
        "Vietnamesischer Dong",
        "displayName-count-one";
        "Vietnamesischer Dong",
        "displayName-count-other";
        "Vietnamesische Dong",
        "symbol";
        "₫",
        "symbol-alt-narrow";
        "₫";
    }
    "VNN";
    {
        "displayName";
        "Vietnamesischer Dong (1978-1985)",
        "displayName-count-one";
        "Vietnamesischer Dong (1978-1985)",
        "displayName-count-other";
        "Vietnamesische Dong (1978-1985)",
        "symbol";
        "VNN";
    }
    "VUV";
    {
        "displayName";
        "Vanuatu-Vatu",
        "displayName-count-one";
        "Vanuatu-Vatu",
        "displayName-count-other";
        "Vanuatu-Vatu",
        "symbol";
        "VUV";
    }
    "WST";
    {
        "displayName";
        "Samoanischer Tala",
        "displayName-count-one";
        "Samoanischer Tala",
        "displayName-count-other";
        "Samoanische Tala",
        "symbol";
        "WST";
    }
    "XAF";

```

```

        {
            "displayName";
            "CFA-Franc (BEAC)",
            "displayName-count-one";
            "CFA-Franc (BEAC)",
            "displayName-count-other";
            "CFA-Franc (BEAC)",
            "symbol";
            "FCFA";
        }
        "XAG";
        {
            "displayName";
            "Unze Silber",
            "displayName-count-one";
            "Unze Silber",
            "displayName-count-other";
            "Unzen Silber",
            "symbol";
            "XAG";
        }
        "XAU";
        {
            "displayName";
            "Unze Gold",
            "displayName-count-one";
            "Unze Gold",
            "displayName-count-other";
            "Unzen Gold",
            "symbol";
            "XAU";
        }
        "XBA";
        {
            "displayName";
            "Europäische Rechnungseinheit",
            "displayName-count-one";
            "Europäische Rechnungseinheiten",
            "displayName-count-other";
            "Europäische Rechnungseinheiten",
            "symbol";
            "XBA";
        }
        "XBB";
        {
            "displayName";
            "Europäische Währungseinheit (XBB)",
            "displayName-count-one";
            "Europäische Währungseinheiten (XBB)",
            "displayName-count-other";
            "Europäische Währungseinheiten (XBB)",
            "symbol";
            "XBB";
        }
        "XBC";
        {
            "displayName";

```

```

        "Europäische Rechnungseinheit (XBC)",
        "displayName-count-one";
        "Europäische Rechnungseinheiten (XBC)",
        "displayName-count-other";
        "Europäische Rechnungseinheiten (XBC)",
        "symbol";
        "XBC";
    }
    "XBD";
    {
        "displayName";
        "Europäische Rechnungseinheit (XBD)",
        "displayName-count-one";
        "Europäische Rechnungseinheiten (XBD)",
        "displayName-count-other";
        "Europäische Rechnungseinheiten (XBD)",
        "symbol";
        "XBD";
    }
    "XCD";
    {
        "displayName";
        "Ostkaribischer Dollar",
        "displayName-count-one";
        "Ostkaribischer Dollar",
        "displayName-count-other";
        "Ostkaribische Dollar",
        "symbol";
        "EC$";
        "symbol-alt-narrow";
        "$";
    }
    "XDR";
    {
        "displayName";
        "Sonderziehungsrechte",
        "displayName-count-one";
        "Sonderziehungsrechte",
        "displayName-count-other";
        "Sonderziehungsrechte",
        "symbol";
        "XDR";
    }
    "XEU";
    {
        "displayName";
        "Europäische Währungseinheit (XEU)",
        "displayName-count-one";
        "Europäische Währungseinheiten (XEU)",
        "displayName-count-other";
        "Europäische Währungseinheiten (XEU)",
        "symbol";
        "XEU";
    }
    "XFO";
    {
        "displayName";

```

```

        "Französischer Gold-Franc",
        "displayName-count-one";
        "Französische Gold-Franc",
        "displayName-count-other";
        "Französische Gold-Franc",
        "symbol";
        "XFO";
    }
    "XFU";
    {
        "displayName";
        "Französischer UIC-Franc",
        "displayName-count-one";
        "Französische UIC-Franc",
        "displayName-count-other";
        "Französische UIC-Franc",
        "symbol";
        "XFU";
    }
    "XOF";
    {
        "displayName";
        "CFA-Franc (BCEAO)",
        "displayName-count-one";
        "CFA-Franc (BCEAO)",
        "displayName-count-other";
        "CFA-Francs (BCEAO)",
        "symbol";
        "CFA";
    }
    "XPD";
    {
        "displayName";
        "Unze Palladium",
        "displayName-count-one";
        "Unze Palladium",
        "displayName-count-other";
        "Unzen Palladium",
        "symbol";
        "XPD";
    }
    "XPF";
    {
        "displayName";
        "CFP-Franc",
        "displayName-count-one";
        "CFP-Franc",
        "displayName-count-other";
        "CFP-Franc",
        "symbol";
        "CFPF";
    }
    "XPT";
    {
        "displayName";
        "Unze Platin",
        "displayName-count-one";

```

```

        "Unze Platin",
        "displayName-count-other";
        "Unzen Platin",
        "symbol";
        "XPT";
    }
    "XRE";
    {
        "displayName";
        "RINET Funds",
        "displayName-count-one";
        "RINET Funds",
        "displayName-count-other";
        "RINET Funds",
        "symbol";
        "XRE";
    }
    "XSU";
    {
        "displayName";
        "SUCRE",
        "displayName-count-one";
        "SUCRE",
        "displayName-count-other";
        "SUCRE",
        "symbol";
        "XSU";
    }
    "XTS";
    {
        "displayName";
        "Testwährung",
        "displayName-count-one";
        "Testwährung",
        "displayName-count-other";
        "Testwährung",
        "symbol";
        "XTS";
    }
    "XUA";
    {
        "displayName";
        "Rechnungseinheit der AfEB",
        "displayName-count-one";
        "Rechnungseinheit der AfEB",
        "displayName-count-other";
        "Rechnungseinheiten der AfEB",
        "symbol";
        "XUA";
    }
    "XXX";
    {
        "displayName";
        "Unbekannte Währung",
        "displayName-count-one";
        "(unbekannte Währung)",
        "displayName-count-other";
    }

```

```

        "(unbekannte Währung)",
        "symbol";
        "XXX";
    }
    "YDD";
    {
        "displayName";
        "Jemen-Dinar",
        "displayName-count-one";
        "Jemen-Dinar",
        "displayName-count-other";
        "Jemen-Dinar",
        "symbol";
        "YDD";
    }
    "YER";
    {
        "displayName";
        "Jemen-Rial",
        "displayName-count-one";
        "Jemen-Rial",
        "displayName-count-other";
        "Jemen-Rial",
        "symbol";
        "YER";
    }
    "YUD";
    {
        "displayName";
        "Jugoslawischer Dinar (1966-1990)",
        "displayName-count-one";
        "Jugoslawischer Dinar (1966-1990)",
        "displayName-count-other";
        "Jugoslawische Dinar (1966-1990)",
        "symbol";
        "YUD";
    }
    "YUM";
    {
        "displayName";
        "Jugoslawischer Neuer Dinar (1994-2002)",
        "displayName-count-one";
        "Jugoslawischer Neuer Dinar (1994-2002)",
        "displayName-count-other";
        "Jugoslawische Neue Dinar (1994-2002)",
        "symbol";
        "YUM";
    }
    "YUN";
    {
        "displayName";
        "Jugoslawischer Dinar (konvertibel)",
        "displayName-count-one";
        "Jugoslawische Dinar (konvertibel)",
        "displayName-count-other";
        "Jugoslawische Dinar (konvertibel)",
        "symbol";
    }

```

```

        "YUN";
    }
    "YUR";
    {
        "displayName";
        "Jugoslawischer reformierter Dinar (1992-1993)",
        "displayName-count-one";
        "Jugoslawischer reformierter Dinar (1992-1993)",
        "displayName-count-other";
        "Jugoslawische reformierte Dinar (1992-1993)",
        "symbol";
        "YUR";
    }
    "ZAL";
    {
        "displayName";
        "Südafrikanischer Rand (Finanz)",
        "displayName-count-one";
        "Südafrikanischer Rand (Finanz)",
        "displayName-count-other";
        "Südafrikanischer Rand (Finanz)",
        "symbol";
        "ZAL";
    }
    "ZAR";
    {
        "displayName";
        "Südafrikanischer Rand",
        "displayName-count-one";
        "Südafrikanischer Rand",
        "displayName-count-other";
        "Südafrikanische Rand",
        "symbol";
        "ZAR",
        "symbol-alt-narrow";
        "R";
    }
    "ZMK";
    {
        "displayName";
        "Kwacha (1968-2012)",
        "displayName-count-one";
        "Kwacha (1968-2012)",
        "displayName-count-other";
        "Kwacha (1968-2012)",
        "symbol";
        "ZMK";
    }
    "ZMW";
    {
        "displayName";
        "Kwacha",
        "displayName-count-one";
        "Kwacha",
        "displayName-count-other";
        "Kwacha",
        "symbol";
    }

```

```

        "ZMW",
        "symbol-alt-narrow";
        "K";
    }
    "ZRN";
    {
        "displayName";
        "Zaire-Neuer Zaïre (1993-1998)",
        "displayName-count-one";
        "Zaire-Neuer Zaïre (1993-1998)",
        "displayName-count-other";
        "Zaire-Neue Zaïre (1993-1998)",
        "symbol";
        "ZRN";
    }
    "ZRZ";
    {
        "displayName";
        "Zaire-Zaïre (1971-1993)",
        "displayName-count-one";
        "Zaire-Zaïre (1971-1993)",
        "displayName-count-other";
        "Zaire-Zaïre (1971-1993)",
        "symbol";
        "ZRZ";
    }
    "ZWD";
    {
        "displayName";
        "Simbabwe-Dollar (1980-2008)",
        "displayName-count-one";
        "Simbabwe-Dollar (1980-2008)",
        "displayName-count-other";
        "Simbabwe-Dollar (1980-2008)",
        "symbol";
        "ZWD";
    }
    "ZWL";
    {
        "displayName";
        "Simbabwe-Dollar (2009)",
        "displayName-count-one";
        "Simbabwe-Dollar (2009)",
        "displayName-count-other";
        "Simbabwe-Dollar (2009)",
        "symbol";
        "ZWL";
    }
    "ZWR";
    {
        "displayName";
        "Simbabwe-Dollar (2008)",
        "displayName-count-one";
        "Simbabwe-Dollar (2008)",
        "displayName-count-other";
        "Simbabwe-Dollar (2008)",
        "symbol";
    }

```



```
}  
}  
}  
}  
}  
} "ZWR";
```

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
//import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import { loadCldr, L10n } from '@syncfusion/ej2-base';
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as timeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames);
L10n.load({
  'de': {
    'calendar': { today: 'heute' }
  }
});
class App extends React.Component {
  render() {
    return <CalendarComponent id="calendar" locale='de'/>;
  }
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
//import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import { loadCldr, L10n } from '@syncfusion/ej2-base';
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as timeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames);
L10n.load({
    'de': {
        'calendar': { today: 'heute' }
    }
});
class App extends React.Component<{}, {}> {
    render() {
        return <CalendarComponent id="calendar" locale='de' />
    }
}
```

```
};
ReactDOM.render(<App />, document.getElementById('element'));
```

NUMBERINGSYSTEMS.JSON

```
{
  "supplemental": {
    "version": {
      "_number": "$Revision: 12732 $",
      "_unicodeVersion": "9.0.0",
      "_cldrVersion": "31"
    },
    "numberingSystems": {
      "adlm": {
        "_digits": "ᲠᲡᲢᲣᲤᲥᲦᲧᲨᲩᲪᲫᲬᲭᲮᲯᲰᲱᲲᲳᲴᲵᲶᲷᲸᲹᲺ᲻᲼ᲽᲾᲿ",
        "_type": "numeric"
      },
      "ahom": {
        "_digits": "ᩌᩍᩎᩏᩐᩑᩒᩓᩔᩕᩖᩗᩘᩙᩚᩛᩜᩝᩞ᩟᩠ᩡᩢᩣᩤᩥᩦᩧᩨᩩᩪᩫᩬᩭᩮᩯᩰᩱᩲᩳᩴ᩵᩶᩷᩸᩹᩺᩻᩼᩽᩾᩿",
        "_type": "numeric"
      },
      "arab": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "arabext": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "armn": {
        "_rules": "armenian-upper",
        "_type": "algorithmic"
      },
      "armnlow": {
        "_rules": "armenian-lower",
        "_type": "algorithmic"
      },
      "bali": {
        "_digits": "ᬀᬁᬂᬃᬄᬅᬆᬇᬈᬉᬊᬋᬌᬍᬎᬏᬐᬑᬒᬓᬔᬕᬖᬗᬘᬙᬚᬛᬜᬝᬞᬟᬠᬡᬢᬣᬤᬥᬦᬧᬨᬩᬪᬫᬬᬭᬮᬯᬰᬱᬲᬳ᬴ᬵᬶᬷᬸᬹᬺᬻᬼᬽᬾᬿ",
        "_type": "numeric"
      },
      "beng": {
        "_digits": "০১২৩৪৫৬৭৮৯",
        "_type": "numeric"
      },
      "bhks": {
        "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍᱎᱏ᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱯᱰᱱᱲᱳᱴᱵᱶᱷᱸᱹᱺᱻᱼᱽ᱾᱿",
        "_type": "numeric"
      },
      "brah": {
        "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍᱎᱏ᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱯᱰᱱᱲᱳᱴᱵᱶᱷᱸᱹᱺᱻᱼᱽ᱾᱿",
        "_type": "numeric"
      },
      "cakm": {
        "_digits": "᩠ᩡᩢᩣᩤᩥᩦᩧᩨᩩᩪᩫᩬᩭᩮᩯᩰᩱᩲᩳᩴ᩵᩶᩷᩸᩹᩺᩻᩼᩽᩾᩿",
        "_type": "numeric"
      }
    }
  }
}
```

```

    "_type": "numeric"
  },
  "cham": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "cyr1": {
    "_rules": "cyrillic-lower",
    "_type": "algorithmic"
  },
  "deva": {
    "_digits": "ॐ३३४५६७८९",
    "_type": "numeric"
  },
  "ethi": {
    "_rules": "ethiopic",
    "_type": "algorithmic"
  },
  "fullwide": {
    "_digits": "0 1 2 3 4 5 6 7 8 9",
    "_type": "numeric"
  },
  "geor": {
    "_rules": "georgian",
    "_type": "algorithmic"
  },
  "grek": {
    "_rules": "greek-upper",
    "_type": "algorithmic"
  },
  "greklow": {
    "_rules": "greek-lower",
    "_type": "algorithmic"
  },
  "gujr": {
    "_digits": "ૐ૩૪૫૬૭૮૯",
    "_type": "numeric"
  },
  "guru": {
    "_digits": "੐੩੪੫੬੭੮੯",
    "_type": "numeric"
  },
  "hanidays": {
    "_rules": "zh/SpelloutRules/spellout-numbering-days",
    "_type": "algorithmic"
  },
  "hanidec": {
    "_digits": "〇一二三四五六七八九",
    "_type": "numeric"
  },
  "hans": {
    "_rules": "zh/SpelloutRules/spellout-cardinal",
    "_type": "algorithmic"
  },
  "hansfin": {
    "_rules": "zh/SpelloutRules/spellout-cardinal-financial",

```

```

    "_type": "algorithmic"
  },
  "hant": {
    "_rules": "zh_Hant/SpelloutRules/spellout-cardinal",
    "_type": "algorithmic"
  },
  "hantfin": {
    "_rules": "zh_Hant/SpelloutRules/spellout-cardinal-financial",
    "_type": "algorithmic"
  },
  "hebr": {
    "_rules": "hebrew",
    "_type": "algorithmic"
  },
  "hmng": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "java": {
    "_digits": "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉᐊᐋᐌᐍᐎᐏᐐᐑᐒᐓᐔᐕᐖᐗᐘᐙᐚᐛᐜᐝᐞᐟᐠᐡᐢᐣᐤᐥᐦᐧᐨᐩᐪᐫᐬᐭᐮᐯᐰᐱᐲᐳᐴᐵᐶᐷᐸᐹᐺᐻᐼᐽᐾᐿ",
    "_type": "numeric"
  },
  "jpan": {
    "_rules": "ja/SpelloutRules/spellout-cardinal",
    "_type": "algorithmic"
  },
  "jpanfin": {
    "_rules": "ja/SpelloutRules/spellout-cardinal-financial",
    "_type": "algorithmic"
  },
  "kali": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "khmr": {
    "_digits": "០១២៣៤៥៦៧៨៩",
    "_type": "numeric"
  },
  "knda": {
    "_digits": "೦೧೨೩೪೫೬೭೮೯",
    "_type": "numeric"
  },
  "lana": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "lanatham": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "laoo": {
    "_digits": "໐໑໒໓໔໕໖໗໘໙",
    "_type": "numeric"
  },
  "latn": {

```

```

    "_digits": "0123456789",
    "_type": "numeric"
  },
  "lepc": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "limb": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "mathbold": {
    "_digits": ""0123456789"",
    "_type": "numeric"
  },
  "mathdbl": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mathmono": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mathsanb": {
    "_digits": ""0123456789"",
    "_type": "numeric"
  },
  "mathsans": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mlym": {
    "_digits": "ഐഘം൧൨൩൪൫൬൭൮൯ൺ",
    "_type": "numeric"
  },
  "modi": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "mong": {
    "_digits": "᠐᠑᠒᠓᠐ᠠᠨᠨᠠᠭᠤᠯᠤᠰ",
    "_type": "numeric"
  },
  "mroo": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "mtei": {
    "_digits": "᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚ",
    "_type": "numeric"
  },
  "mymr": {
    "_digits": "၀၁၂၃၄၅၆၇၈၉",
    "_type": "numeric"
  },
  },

```

```

    "mymrshan": {
      "_digits": "0123456789",
      "_type": "numeric"
    },
    "mymrtlng": {
      "_digits": "0၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "newa": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "nkoo": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "olck": {
      "_digits": "0123456789",
      "_type": "numeric"
    },
    "orya": {
      "_digits": "୦୧୨୩୪୫୬୭୮୯",
      "_type": "numeric"
    },
    "osma": {
      "_digits": "0ᲠᲡᲢᲣᲤᲥᲦᲧᲨᲩ",
      "_type": "numeric"
    },
    "roman": {
      "_rules": "roman-upper",
      "_type": "algorithmic"
    },
    "romanlow": {
      "_rules": "roman-lower",
      "_type": "algorithmic"
    },
    "saur": {
      "_digits": "0123456789",
      "_type": "numeric"
    },
    "shrd": {
      "_digits": "0123456789",
      "_type": "numeric"
    },
    "sind": {
      "_digits": "0123456789",
      "_type": "numeric"
    },
    "sinh": {
      "_digits": "୦୧୨୩୪୫୬୭୮୯",
      "_type": "numeric"
    },
    "sora": {
      "_digits": "0ᲠᲡᲢᲣᲤᲥᲦᲧᲨᲩ",

```

```

    "_type": "numeric"
  },
  "sund": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "takr": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "talv": {
    "_digits": "တၢ်ၣ်တၢ်တၢ်တၢ်",
    "_type": "numeric"
  },
  "taml": {
    "_rules": "tamil",
    "_type": "algorithmic"
  },
  "tamldec": {
    "_digits": "0௧௨௩௪௫௬௭௮௯",
    "_type": "numeric"
  },
  "telu": {
    "_digits": "౦౧౨౩౪౫౬౭౮౯",
    "_type": "numeric"
  },
  "thai": {
    "_digits": "๐๑๒๓๔๕๖๗๘๙",
    "_type": "numeric"
  },
  "tibb": {
    "_digits": "༠༡༢༣༤༥༦༧༨༩",
    "_type": "numeric"
  },
  "tirh": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "vaih": {
    "_digits": "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
    "_type": "numeric"
  },
  "wara": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  }
}
}
}

```

NUMBERINGSYSTEMS.JSX

```

{
  "supplemental";
  {

```

```

"version";
{
  "_number";
  "$Revision: 12732 $",
  "_unicodeVersion";
  "9.0.0",
  "_cldrVersion";
  "31";
}
"numberingSystems";
{
  "adlm";
  {
    "_digits";
    "ᲐᲑᲒᲓᲔᲕᲖᲗᲘᲙ",
    "_type";
    "numeric";
  }
  "ahom";
  {
    "_digits";
    "ᩌᩍᩎᩏᩐᩑᩒᩓᩔᩕᩖ",
    "_type";
    "numeric";
  }
  "arab";
  {
    "_digits";
    "٠١٢٣٤٥٦٧٨٩",
    "_type";
    "numeric";
  }
  "arabext";
  {
    "_digits";
    "٠١٢٣٤٥٦٧٨٩",
    "_type";
    "numeric";
  }
  "armn";
  {
    "_rules";
    "armenian-upper",
    "_type";
    "algorithmic";
  }
  "armnlow";
  {
    "_rules";
    "armenian-lower",
    "_type";
    "algorithmic";
  }
  "bali";
  {
    "_digits";

```



```

        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "beng";
    {
        "_digits";
        "০১২৩৪৫৬৭৮৯",
        "_type";
        "numeric";
    }
    "bhks";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "brah";
    {
        "_digits";
        "·\u094d\u0947\u0940",
        "_type";
        "numeric";
    }
    "cakm";
    {
        "_digits";
        "ᱠᱡᱣᱤᱨᱢᱤᱰᱤᱦ",
        "_type";
        "numeric";
    }
    "cham";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "cyrl";
    {
        "_rules";
        "cyrillic-lower",
        "_type";
        "algorithmic";
    }
    "deva";
    {
        "_digits";
        "०१२३४५६७८९",
        "_type";
        "numeric";
    }
    "ethi";
    {
        "_rules";

```

```

        "ethiopic",
        "_type";
        "algorithmic";
    }
    "fullwide";
    {
        "_digits";
        "0 1 2 3 4 5 6 7 8 9",
        "_type";
        "numeric";
    }
    "geor";
    {
        "_rules";
        "georgian",
        "_type";
        "algorithmic";
    }
    "grek";
    {
        "_rules";
        "greek-upper",
        "_type";
        "algorithmic";
    }
    "greklow";
    {
        "_rules";
        "greek-lower",
        "_type";
        "algorithmic";
    }
    "gujr";
    {
        "_digits";
        "૦ ૧ ૨ ૩ ૪ ૫ ૬ ૭ ૮ ૯",
        "_type";
        "numeric";
    }
    "guru";
    {
        "_digits";
        "੦ ੧ ੨ ੩ ੪ ੫ ੬ ੭ ੮ ੯",
        "_type";
        "numeric";
    }
    "hanidays";
    {
        "_rules";
        "zh/SpelloutRules/spellout-numbering-days",
        "_type";
        "algorithmic";
    }
    "hanidec";
    {
        "_digits";

```

```
"〇一三四五六七八九",
    "_type";
    "numeric";
}
"hans";
{
    "_rules";
    "zh/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
}
"hansfin";
{
    "_rules";
    "zh/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
}
"hant";
{
    "_rules";
    "zh_Hant/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
}
"hantfin";
{
    "_rules";
    "zh_Hant/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
}
"hebr";
{
    "_rules";
    "hebrew",
        "_type";
        "algorithmic";
}
"hmng";
{
    "_digits";
    "᠐᠑᠒᠓᠔᠕᠖᠗᠘᠙",
        "_type";
        "numeric";
}
"java";
{
    "_digits";
    "᠐ᠠᠨᠵᠢᠮᠤᠩᠭᠡᠳᠦ᠋ᠨ",
        "_type";
        "numeric";
}
"jpan";
{
```

```

        "_rules";
        "ja/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "jpanfin";
    {
        "_rules";
        "ja/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "kali";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
    "khmr";
    {
        "_digits";
        "០១២៣៤៥៦៧៨៩",
        "_type";
        "numeric";
    }
    "knda";
    {
        "_digits";
        "೦೧೨೩೪೫೬೭೮೯",
        "_type";
        "numeric";
    }
    "lana";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
    "lanatham";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
    "laoo";
    {
        "_digits";
        "໐໑໒໓໔໕໖໗໘໑",
        "_type";
        "numeric";
    }
    "latn";
    {

```

```

        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "lepc";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "limb";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "mathbold";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathdbl";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathmono";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathsanb";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathsans";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mlym";
    {

```

```

        "_digits";
        "൧൨൩൪൫൬൭൮൯",
        "_type";
        "numeric";
    }
    "modi";
    {
        "_digits";
        "൦൧൨൩൪൫൬൭൮൯",
        "_type";
        "numeric";
    }
    "mong";
    {
        "_digits";
        "᠐᠑᠒᠓᠔᠕᠖᠗᠘᠐",
        "_type";
        "numeric";
    }
    "mroo";
    {
        "_digits";
        "ᦅᦺᦑᦟᦹᦺᦑᦟᦹᦺ",
        "_type";
        "numeric";
    }
    "mtei";
    {
        "_digits";
        "ႤႬႬႬႬႬႬႬႬႬႬႬ",
        "_type";
        "numeric";
    }
    "mymr";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "mymrshan";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "mymrtlng";
    {
        "_digits";
        "ᦅᦺᦑᦟᦹᦺᦑᦟᦹᦺ",
        "_type";
        "numeric";
    }

```

```

"newa";
{
    "_digits";
    "□□□□□□□□□□",
    "_type";
    "numeric";
}
"nkoo";
{
    "_digits";
    "ᲚᲛᲞᲟᲠᲡᲢᲣᲤᲥ",
    "_type";
    "numeric";
}
"olck";
{
    "_digits";
    "ᲚᲛᲞᲟᲠᲡᲢᲣᲤᲥ",
    "_type";
    "numeric";
}
"orya";
{
    "_digits";
    "ᲚᲛᲞᲟᲠᲡᲢᲣᲤᲥ",
    "_type";
    "numeric";
}
"osma";
{
    "_digits";
    "ᲚᲛᲞᲟᲠᲡᲢᲣᲤᲥ",
    "_type";
    "numeric";
}
"roman";
{
    "_rules";
    "roman-upper",
    "_type";
    "algorithmic";
}
"romanlow";
{
    "_rules";
    "roman-lower",
    "_type";
    "algorithmic";
}
"saur";
{
    "_digits";
    "□□□□□□□□□□",
    "_type";
    "numeric";
}

```

```

"shrd";
{
  "_digits";
  "□□□□□□□□",
  "_type";
  "numeric";
}
"sind";
{
  "_digits";
  "□□□□□□□□",
  "_type";
  "numeric";
}
"sinh";
{
  "_digits";
  "⁂௮௮௮௮௮௮௮௮",
  "_type";
  "numeric";
}
"sora";
{
  "_digits";
  "0௧௨௩௪௫௬௭௮",
  "_type";
  "numeric";
}
"sund";
{
  "_digits";
  "□□□□□□□□",
  "_type";
  "numeric";
}
"takr";
{
  "_digits";
  "□□□□□□□□",
  "_type";
  "numeric";
}
"talv";
{
  "_digits";
  "ᱠᱡᱣᱤᱨᱢᱟ",
  "_type";
  "numeric";
}
"taml";
{
  "_rules";
  "tamil",
  "_type";
  "algorithmic";
}

```



```

    "tamldec";
    {
      "_digits";
      "௦௧௨௩௪௫௬௭௮௯",
      "_type";
      "numeric";
    }
    "telu";
    {
      "_digits";
      "౦౧౨౩౪౫౬౭౮౯",
      "_type";
      "numeric";
    }
    "thai";
    {
      "_digits";
      "๐๑๒๓๔๕๖๗๘๙",
      "_type";
      "numeric";
    }
    "tibb";
    {
      "_digits";
      "༠༡༢༣༤༥༦༧༨༩",
      "_type";
      "numeric";
    }
    "tirh";
    {
      "_digits";
      "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉",
      "_type";
      "numeric";
    }
    "vaih";
    {
      "_digits";
      "᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙",
      "_type";
      "numeric";
    }
    "wara";
    {
      "_digits";
      "ᳵᳶ᳷᳸᳹ᳺ᳻᳼᳽᳾᳿",
      "_type";
      "numeric";
    }
  }
}

```

NUMBERS.JSON

```

{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31"
        },
        "language": "de"
      },
      "numbers": {
        "defaultNumberingSystem": "latn",
        "otherNumberingSystems": {
          "native": "latn"
        },
        "minimumGroupingDigits": "1",
        "symbols-numberSystem-latn": {
          "decimal": ",",
          "group": ".",
          "list": ";",
          "percentSign": "%",
          "plusSign": "+",
          "minusSign": "-",
          "exponential": "E",
          "superscriptingExponent": "·",
          "perMille": "‰",
          "infinity": "∞",
          "nan": "NaN",
          "timeSeparator": ":"
        },
        "decimalFormats-numberSystem-latn": {
          "standard": "#,##0.###",
          "long": {
            "decimalFormat": {
              "1000-count-one": "0 Tausend",
              "1000-count-other": "0 Tausend",
              "10000-count-one": "00 Tausend",
              "10000-count-other": "00 Tausend",
              "100000-count-one": "000 Tausend",
              "100000-count-other": "000 Tausend",
              "1000000-count-one": "0 Million",
              "1000000-count-other": "0 Millionen",
              "10000000-count-one": "00 Millionen",
              "10000000-count-other": "00 Millionen",
              "100000000-count-one": "000 Millionen",
              "100000000-count-other": "000 Millionen",
              "1000000000-count-one": "0 Milliarde",
              "1000000000-count-other": "0 Milliarden",
              "10000000000-count-one": "00 Milliarden",
              "10000000000-count-other": "00 Milliarden",
              "100000000000-count-one": "000 Milliarden",
              "100000000000-count-other": "000 Milliarden",
              "1000000000000-count-one": "0 Billion",
              "1000000000000-count-other": "0 Billionen",
              "10000000000000-count-one": "00 Billionen",
              "10000000000000-count-other": "00 Billionen",
              "100000000000000-count-one": "000 Billionen",
              "100000000000000-count-other": "000 Billionen"
            }
          }
        }
      }
    }
  }
}

```

```

        "1000000000000000-count-other": "000 Billionen"
    }
},
"short": {
    "decimalFormat": {
        "1000-count-one": "0",
        "1000-count-other": "0",
        "10000-count-one": "0",
        "10000-count-other": "0",
        "100000-count-one": "0",
        "100000-count-other": "0",
        "1000000-count-one": "0 Mio'.'",
        "1000000-count-other": "0 Mio'.'",
        "10000000-count-one": "00 Mio'.'",
        "10000000-count-other": "00 Mio'.'",
        "100000000-count-one": "000 Mio'.'",
        "100000000-count-other": "000 Mio'.'",
        "1000000000-count-one": "0 Mrd'.'",
        "1000000000-count-other": "0 Mrd'.'",
        "10000000000-count-one": "00 Mrd'.'",
        "10000000000-count-other": "00 Mrd'.'",
        "100000000000-count-one": "000 Mrd'.'",
        "100000000000-count-other": "000 Mrd'.'",
        "1000000000000-count-one": "0 Bio'.'",
        "1000000000000-count-other": "0 Bio'.'",
        "10000000000000-count-one": "00 Bio'.'",
        "10000000000000-count-other": "00 Bio'.'",
        "100000000000000-count-one": "000 Bio'.'",
        "100000000000000-count-other": "000 Bio'.'"
    }
}
},
"scientificFormats-numberSystem-latn": {
    "standard": "#E0"
},
"percentFormats-numberSystem-latn": {
    "standard": "#,##0 %"
},
"currencyFormats-numberSystem-latn": {
    "currencySpacing": {
        "beforeCurrency": {
            "currencyMatch": "[:^S:]",
            "surroundingMatch": "[:digit:]",
            "insertBetween": " "
        },
        "afterCurrency": {
            "currencyMatch": "[:^S:]",
            "surroundingMatch": "[:digit:]",
            "insertBetween": " "
        }
    },
    "standard": "#,##0.00 ¤",
    "accounting": "#,##0.00 ¤",
    "short": {
        "standard": {
            "1000-count-one": "0 Tsd'.' ¤",
            "1000-count-other": "0 Tsd'.' ¤",

```

```

    "10000-count-one": "00 Tsd'.' α",
    "10000-count-other": "00 Tsd'.' α",
    "100000-count-one": "000 Tsd'.' α",
    "100000-count-other": "000 Tsd'.' α",
    "1000000-count-one": "0 Mio'.' α",
    "1000000-count-other": "0 Mio'.' α",
    "10000000-count-one": "00 Mio'.' α",
    "10000000-count-other": "00 Mio'.' α",
    "100000000-count-one": "000 Mio'.' α",
    "100000000-count-other": "000 Mio'.' α",
    "1000000000-count-one": "0 Mrd'.' α",
    "1000000000-count-other": "0 Mrd'.' α",
    "10000000000-count-one": "00 Mrd'.' α",
    "10000000000-count-other": "00 Mrd'.' α",
    "100000000000-count-one": "000 Mrd'.' α",
    "100000000000-count-other": "000 Mrd'.' α",
    "1000000000000-count-one": "0 Bio'.' α",
    "1000000000000-count-other": "0 Bio'.' α",
    "10000000000000-count-one": "00 Bio'.' α",
    "10000000000000-count-other": "00 Bio'.' α",
    "100000000000000-count-one": "000 Bio'.' α",
    "100000000000000-count-other": "000 Bio'.' α"
  },
},
"unitPattern-count-one": "{0} {1}",
"unitPattern-count-other": "{0} {1}"
},
"miscPatterns-numberSystem-latn": {
  "atLeast": "{0}+",
  "range": "{0}-{1}"
},
"minimalPairs": {
  "pluralMinimalPairs": "{0} Tag",
  "pluralMinimalPairs": "{0} Tage",
  "other": "{0}. Abzweigung nach rechts nehmen"
}
}
}
}
```

NUMBERS.JSX

```
{  
    "main";  
    {  
        "de";  
        {  
            "identity";  
            {  
                "version";  
                {  
                    "_number";  
                    "$Revision: 13259 $",  
                        "_cldrVersion";  
                        "31";
```

```

    }
    "language";
    "de";
  }
  "numbers";
  {
    "defaultNumberingSystem";
    "latn",
    "otherNumberingSystems";
    {
      "native";
      "latn";
    }
    "minimumGroupingDigits";
    "1",
    "symbols-numberSystem-latn";
    {
      "decimal";
      ",",
      "group";
      ".",
      "list";
      ";",
      "percentSign";
      "%",
      "plusSign";
      "+",
      "minusSign";
      "-",
      "exponential";
      "E",
      "superscriptingExponent";
      ". ",
      "perMille";
      "‰",
      "infinity";
      "∞",
      "nan";
      "NaN",
      "timeSeparator";
      ":";
    }
    "decimalFormats-numberSystem-latn";
    {
      "standard";
      "#,##0.###",
      "long";
      {
        "decimalFormat";
        {
          "1000-count-one";
          "0 Tausend",
          "1000-count-other";
          "0 Tausend",
          "10000-count-one";
          "00 Tausend",
          "10000-count-other";
        }
      }
    }
  }

```

```

        "00 Tausend",
        "100000-count-one";
        "000 Tausend",
        "100000-count-other";
        "000 Tausend",
        "1000000-count-one";
        "0 Million",
        "1000000-count-other";
        "0 Millionen",
        "10000000-count-one";
        "00 Millionen",
        "10000000-count-other";
        "00 Millionen",
        "100000000-count-one";
        "000 Millionen",
        "100000000-count-other";
        "000 Millionen",
        "1000000000-count-one";
        "0 Milliarde",
        "1000000000-count-other";
        "0 Milliarden",
        "10000000000-count-one";
        "00 Milliarden",
        "10000000000-count-other";
        "00 Milliarden",
        "100000000000-count-one";
        "000 Milliarden",
        "100000000000-count-other";
        "000 Milliarden",
        "1000000000000-count-one";
        "0 Billion",
        "1000000000000-count-other";
        "0 Billionen",
        "10000000000000-count-one";
        "00 Billionen",
        "10000000000000-count-other";
        "00 Billionen",
        "100000000000000-count-one";
        "000 Billionen",
        "100000000000000-count-other";
        "000 Billionen";
    }
}
"short";
{
    "decimalFormat";
    {
        "1000-count-one";
        "0",
        "1000-count-other";
        "0",
        "10000-count-one";
        "0",
        "10000-count-other";
        "0",
        "100000-count-one";
        "0",
    }
}

```

```

        "100000-count-other";
        "0",
        "1000000-count-one";
        "0 Mio'.'",
        "1000000-count-other";
        "0 Mio'.'",
        "10000000-count-one";
        "00 Mio'.'",
        "10000000-count-other";
        "00 Mio'.'",
        "100000000-count-one";
        "000 Mio'.'",
        "100000000-count-other";
        "000 Mio'.'",
        "1000000000-count-one";
        "0 Mrd'.'",
        "1000000000-count-other";
        "0 Mrd'.'",
        "10000000000-count-one";
        "00 Mrd'.'",
        "10000000000-count-other";
        "00 Mrd'.'",
        "100000000000-count-one";
        "000 Mrd'.'",
        "100000000000-count-other";
        "000 Mrd'.'",
        "1000000000000-count-one";
        "0 Bio'.'",
        "1000000000000-count-other";
        "0 Bio'.'",
        "10000000000000-count-one";
        "00 Bio'.'",
        "10000000000000-count-other";
        "00 Bio'.'",
        "100000000000000-count-one";
        "000 Bio'.'",
        "100000000000000-count-other";
        "000 Bio'.'";
    }
}
}
"scientificFormats-numberSystem-latn";
{
    "standard";
    "#E0";
}
"percentFormats-numberSystem-latn";
{
    "standard";
    "#,##0 %";
}
"currencyFormats-numberSystem-latn";
{
    "currencySpacing";
    {
        "beforeCurrency";
        {

```

```

        "currencyMatch";
        "[:^S:]",
        "surroundingMatch";
        "[:digit:]",
        "insertBetween";
        " ";
    }
    "afterCurrency";
    {
        "currencyMatch";
        "[:^S:]",
        "surroundingMatch";
        "[:digit:]",
        "insertBetween";
        " ";
    }
}
"standard";
"#,##0.00 ¤",
    "accounting";
"#,##0.00 ¤",
    "short";
{
    "standard";
    {
        "1000-count-one";
        "0 Tsd'.' ¤",
        "1000-count-other";
        "0 Tsd'.' ¤",
        "10000-count-one";
        "00 Tsd'.' ¤",
        "10000-count-other";
        "00 Tsd'.' ¤",
        "100000-count-one";
        "000 Tsd'.' ¤",
        "100000-count-other";
        "000 Tsd'.' ¤",
        "1000000-count-one";
        "0 Mio'.' ¤",
        "1000000-count-other";
        "0 Mio'.' ¤",
        "10000000-count-one";
        "00 Mio'.' ¤",
        "10000000-count-other";
        "00 Mio'.' ¤",
        "100000000-count-one";
        "000 Mio'.' ¤",
        "100000000-count-other";
        "000 Mio'.' ¤",
        "1000000000-count-one";
        "0 Mrd'.' ¤",
        "1000000000-count-other";
        "0 Mrd'.' ¤",
        "10000000000-count-one";
        "00 Mrd'.' ¤",
        "10000000000-count-other";
        "00 Mrd'.' ¤",
    }
}

```



```

        "100000000000-count-one";
        "000 Mrd'.' s",
        "100000000000-count-other";
        "000 Mrd'.' s",
        "100000000000-count-one";
        "0 Bio'.' s",
        "100000000000-count-other";
        "0 Bio'.' s",
        "100000000000-count-one";
        "00 Bio'.' s",
        "100000000000-count-other";
        "00 Bio'.' s",
        "100000000000-count-one";
        "000 Bio'.' s",
        "100000000000-count-other";
        "000 Bio'.' s";
    }
}
"unitPattern-count-one";
"{0} {1}",
    "unitPattern-count-other";
"{0} {1}";
}
"miscPatterns-numberSystem-latn";
{
    "atLeast";
    "{0}+",
        "range";
    "{0}-{1}";
}
"minimalPairs";
{
    "pluralMinimalPairs";
    "{0} Tag",
        "pluralMinimalPairs";
    "{0} Tage",
        "other";
    "{0}. Abzweigung nach rechts nehmen";
}
}
}
}

```

TIMEZONENAMES.JSON

```
{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 12879 $",
          "_cldrVersion": "30.0.3"
        },
        "language": "de"
      }
    }
  }
}
```

```

"dates": {
  "timeZoneNames": {
    "hourFormat": "+HH:mm;-HH:mm",
    "gmtFormat": "GMT{0}",
    "gmtZeroFormat": "GMT",
    "regionFormat": "{0} Zeit",
    "regionFormat-type-daylight": "{0} Sommerzeit",
    "regionFormat-type-standard": "{0} Normalzeit",
    "fallbackFormat": "{1} ({0})",
    "zone": {
      "America": {
        "Adak": {
          "exemplarCity": "Adak"
        },
        "Anchorage": {
          "exemplarCity": "Anchorage"
        },
        "Anguilla": {
          "exemplarCity": "Anguilla"
        },
        "Antigua": {
          "exemplarCity": "Antigua"
        },
        "Araguaina": {
          "exemplarCity": "Araguaina"
        },
        "Argentina": {
          "Rio_Gallegos": {
            "exemplarCity": "Rio Gallegos"
          },
          "San_Juan": {
            "exemplarCity": "San Juan"
          },
          "Ushuaia": {
            "exemplarCity": "Ushuaia"
          },
          "La_Rioja": {
            "exemplarCity": "La Rioja"
          },
          "San_Luis": {
            "exemplarCity": "San Luis"
          },
          "Salta": {
            "exemplarCity": "Salta"
          },
          "Tucuman": {
            "exemplarCity": "Tucuman"
          }
        },
        "Aruba": {
          "exemplarCity": "Aruba"
        },
        "Asuncion": {
          "exemplarCity": "Asunción"
        },
        "Bahia": {
          "exemplarCity": "Bahia"
        }
      }
    }
  }
}

```

```
    },  
    "Bahia_Banderas": {  
      "exemplarCity": "Bahia Banderas"  
    },  
    "Barbados": {  
      "exemplarCity": "Barbados"  
    },  
    "Belem": {  
      "exemplarCity": "Belem"  
    },  
    "Belize": {  
      "exemplarCity": "Belize"  
    },  
    "Blanc-Sablon": {  
      "exemplarCity": "Blanc-Sablon"  
    },  
    "Boa_Vista": {  
      "exemplarCity": "Boa Vista"  
    },  
    "Bogota": {  
      "exemplarCity": "Bogotá"  
    },  
    "Boise": {  
      "exemplarCity": "Boise"  
    },  
    "Buenos_Aires": {  
      "exemplarCity": "Buenos Aires"  
    },  
    "Cambridge_Bay": {  
      "exemplarCity": "Cambridge Bay"  
    },  
    "Campo_Grande": {  
      "exemplarCity": "Campo Grande"  
    },  
    "Cancun": {  
      "exemplarCity": "Cancún"  
    },  
    "Caracas": {  
      "exemplarCity": "Caracas"  
    },  
    "Catamarca": {  
      "exemplarCity": "Catamarca"  
    },  
    "Cayenne": {  
      "exemplarCity": "Cayenne"  
    },  
    "Cayman": {  
      "exemplarCity": "Kaimaninseln"  
    },  
    "Chicago": {  
      "exemplarCity": "Chicago"  
    },  
    "Chihuahua": {  
      "exemplarCity": "Chihuahua"  
    },  
    "Coral_Harbour": {  
      "exemplarCity": "Atikokan"
```

```
    },
    "Cordoba": {
      "exemplarCity": "Córdoba"
    },
    "Costa_Rica": {
      "exemplarCity": "Costa Rica"
    },
    "Creston": {
      "exemplarCity": "Creston"
    },
    "Cuiaba": {
      "exemplarCity": "Cuiaba"
    },
    "Curacao": {
      "exemplarCity": "Curaçao"
    },
    "Danmarkshavn": {
      "exemplarCity": "Danmarkshavn"
    },
    "Dawson": {
      "exemplarCity": "Dawson"
    },
    "Dawson_Creek": {
      "exemplarCity": "Dawson Creek"
    },
    "Denver": {
      "exemplarCity": "Denver"
    },
    "Detroit": {
      "exemplarCity": "Detroit"
    },
    "Dominica": {
      "exemplarCity": "Dominica"
    },
    "Edmonton": {
      "exemplarCity": "Edmonton"
    },
    "Eirunepe": {
      "exemplarCity": "Eirunepe"
    },
    "El_Salvador": {
      "exemplarCity": "El Salvador"
    },
    "Fort_Nelson": {
      "exemplarCity": "Fort Nelson"
    },
    "Fortaleza": {
      "exemplarCity": "Fortaleza"
    },
    "Glace_Bay": {
      "exemplarCity": "Glace Bay"
    },
    "Godthab": {
      "exemplarCity": "Nuuk"
    },
    "Goose_Bay": {
      "exemplarCity": "Goose Bay"
    }
  }
```

```
    },
    "Grand_Turk": {
      "exemplarCity": "Grand Turk"
    },
    "Grenada": {
      "exemplarCity": "Grenada"
    },
    "Guadeloupe": {
      "exemplarCity": "Guadeloupe"
    },
    "Guatemala": {
      "exemplarCity": "Guatemala"
    },
    "Guayaquil": {
      "exemplarCity": "Guayaquil"
    },
    "Guyana": {
      "exemplarCity": "Guyana"
    },
    "Halifax": {
      "exemplarCity": "Halifax"
    },
    "Havana": {
      "exemplarCity": "Havanna"
    },
    "Hermosillo": {
      "exemplarCity": "Hermosillo"
    },
    "Indiana": {
      "Vincennes": {
        "exemplarCity": "Vincennes, Indiana"
      },
      "Petersburg": {
        "exemplarCity": "Petersburg, Indiana"
      },
      "Tell_City": {
        "exemplarCity": "Tell City, Indiana"
      },
      "Knox": {
        "exemplarCity": "Knox, Indiana"
      },
      "Winamac": {
        "exemplarCity": "Winamac, Indiana"
      },
      "Marengo": {
        "exemplarCity": "Marengo, Indiana"
      },
      "Vevay": {
        "exemplarCity": "Vevay, Indiana"
      }
    },
    "Indianapolis": {
      "exemplarCity": "Indianapolis"
    },
    "Inuvik": {
      "exemplarCity": "Inuvik"
    },
  },
```

```
"Iqaluit": {
  "exemplarCity": "Iqaluit"
},
"Jamaica": {
  "exemplarCity": "Jamaika"
},
"Jujuy": {
  "exemplarCity": "Jujuy"
},
"Juneau": {
  "exemplarCity": "Juneau"
},
"Kentucky": {
  "Monticello": {
    "exemplarCity": "Monticello, Kentucky"
  }
},
"Kralendijk": {
  "exemplarCity": "Kralendijk"
},
"La_Paz": {
  "exemplarCity": "La Paz"
},
"Lima": {
  "exemplarCity": "Lima"
},
"Los_Angeles": {
  "exemplarCity": "Los Angeles"
},
"Louisville": {
  "exemplarCity": "Louisville"
},
"Lower_Princes": {
  "exemplarCity": "Lower Prince's Quarter"
},
"Maceio": {
  "exemplarCity": "Maceio"
},
"Managua": {
  "exemplarCity": "Managua"
},
"Manaus": {
  "exemplarCity": "Manaus"
},
"Marigot": {
  "exemplarCity": "Marigot"
},
"Martinique": {
  "exemplarCity": "Martinique"
},
"Matamoros": {
  "exemplarCity": "Matamoros"
},
"Mazatlan": {
  "exemplarCity": "Mazatlan"
},
"Mendoza": {
```

```
        "exemplarCity": "Mendoza"
    },
    "Menominee": {
        "exemplarCity": "Menominee"
    },
    "Merida": {
        "exemplarCity": "Merida"
    },
    "Metlakatla": {
        "exemplarCity": "Metlakatla"
    },
    "Mexico_City": {
        "exemplarCity": "Mexiko-Stadt"
    },
    "Miquelon": {
        "exemplarCity": "Miquelon"
    },
    "Moncton": {
        "exemplarCity": "Moncton"
    },
    "Monterrey": {
        "exemplarCity": "Monterrey"
    },
    "Montevideo": {
        "exemplarCity": "Montevideo"
    },
    "Montserrat": {
        "exemplarCity": "Montserrat"
    },
    "Nassau": {
        "exemplarCity": "Nassau"
    },
    "New_York": {
        "exemplarCity": "New York"
    },
    "Nipigon": {
        "exemplarCity": "Nipigon"
    },
    "Nome": {
        "exemplarCity": "Nome"
    },
    "Noronha": {
        "exemplarCity": "Noronha"
    },
    "North_Dakota": {
        "Beulah": {
            "exemplarCity": "Beulah, North Dakota"
        },
        "New_Salem": {
            "exemplarCity": "New Salem, North Dakota"
        },
        "Center": {
            "exemplarCity": "Center, North Dakota"
        }
    },
    "Ojinaga": {
        "exemplarCity": "Ojinaga"
    }
```

```
    },
    "Panama": {
      "exemplarCity": "Panama"
    },
    "Pangnirtung": {
      "exemplarCity": "Pangnirtung"
    },
    "Paramaribo": {
      "exemplarCity": "Paramaribo"
    },
    "Phoenix": {
      "exemplarCity": "Phoenix"
    },
    "Port-au-Prince": {
      "exemplarCity": "Port-au-Prince"
    },
    "Port_of_Spain": {
      "exemplarCity": "Port of Spain"
    },
    "Porto_Velho": {
      "exemplarCity": "Porto Velho"
    },
    "Puerto_Rico": {
      "exemplarCity": "Puerto Rico"
    },
    "Rainy_River": {
      "exemplarCity": "Rainy River"
    },
    "Rankin_Inlet": {
      "exemplarCity": "Rankin Inlet"
    },
    "Recife": {
      "exemplarCity": "Recife"
    },
    "Regina": {
      "exemplarCity": "Regina"
    },
    "Resolute": {
      "exemplarCity": "Resolute"
    },
    "Rio_Branco": {
      "exemplarCity": "Rio Branco"
    },
    "Santa_Isabel": {
      "exemplarCity": "Santa Isabel"
    },
    "Santarem": {
      "exemplarCity": "Santarem"
    },
    "Santiago": {
      "exemplarCity": "Santiago"
    },
    "Santo_Domingo": {
      "exemplarCity": "Santo Domingo"
    },
    "Sao_Paulo": {
      "exemplarCity": "São Paulo"
    }
  },
}
```



```
    },
    "Scoresbysund": {
      "exemplarCity": "Ittoqqortoormiit"
    },
    "Sitka": {
      "exemplarCity": "Sitka"
    },
    "St_Barthelemy": {
      "exemplarCity": "Saint-Barthélemy"
    },
    "St_Johns": {
      "exemplarCity": "St. John's"
    },
    "St_Kitts": {
      "exemplarCity": "St. Kitts"
    },
    "St_Lucia": {
      "exemplarCity": "St. Lucia"
    },
    "St_Thomas": {
      "exemplarCity": "St. Thomas"
    },
    "St_Vincent": {
      "exemplarCity": "St. Vincent"
    },
    "Swift_Current": {
      "exemplarCity": "Swift Current"
    },
    "Tegucigalpa": {
      "exemplarCity": "Tegucigalpa"
    },
    "Thule": {
      "exemplarCity": "Thule"
    },
    "Thunder_Bay": {
      "exemplarCity": "Thunder Bay"
    },
    "Tijuana": {
      "exemplarCity": "Tijuana"
    },
    "Toronto": {
      "exemplarCity": "Toronto"
    },
    "Tortola": {
      "exemplarCity": "Tortola"
    },
    "Vancouver": {
      "exemplarCity": "Vancouver"
    },
    "Whitehorse": {
      "exemplarCity": "Whitehorse"
    },
    "Winnipeg": {
      "exemplarCity": "Winnipeg"
    },
    "Yakutat": {
      "exemplarCity": "Yakutat"
    }
  }
```

```

    },
    "Yellowknife": {
      "exemplarCity": "Yellowknife"
    }
  },
  "Atlantic": {
    "Azores": {
      "exemplarCity": "Azoren"
    },
    "Bermuda": {
      "exemplarCity": "Bermudas"
    },
    "Canary": {
      "exemplarCity": "Kanaren"
    },
    "Cape_Verde": {
      "exemplarCity": "Cabo Verde"
    },
    "Faeroe": {
      "exemplarCity": "Färöer"
    },
    "Madeira": {
      "exemplarCity": "Madeira"
    },
    "Reykjavik": {
      "exemplarCity": "Reykjavík"
    },
    "South_Georgia": {
      "exemplarCity": "Südgeorgien"
    },
    "St_Helena": {
      "exemplarCity": "St. Helena"
    },
    "Stanley": {
      "exemplarCity": "Stanley"
    }
  },
  "Europe": {
    "Amsterdam": {
      "exemplarCity": "Amsterdam"
    },
    "Andorra": {
      "exemplarCity": "Andorra"
    },
    "Astrakhan": {
      "exemplarCity": "Astrachan"
    },
    "Athens": {
      "exemplarCity": "Athen"
    },
    "Belgrade": {
      "exemplarCity": "Belgrad"
    },
    "Berlin": {
      "exemplarCity": "Berlin"
    },
    "Bratislava": {

```

```
    "exemplarCity": "Bratislava"
  },
  "Brussels": {
    "exemplarCity": "Brüssel"
  },
  "Bucharest": {
    "exemplarCity": "Bukarest"
  },
  "Budapest": {
    "exemplarCity": "Budapest"
  },
  "Busingen": {
    "exemplarCity": "Büsingen"
  },
  "Chisinau": {
    "exemplarCity": "Kischinau"
  },
  "Copenhagen": {
    "exemplarCity": "Kopenhagen"
  },
  "Dublin": {
    "long": {
      "daylight": "Irische Sommerzeit"
    },
    "exemplarCity": "Dublin"
  },
  "Gibraltar": {
    "exemplarCity": "Gibraltar"
  },
  "Guernsey": {
    "exemplarCity": "Guernsey"
  },
  "Helsinki": {
    "exemplarCity": "Helsinki"
  },
  "Isle_of_Man": {
    "exemplarCity": "Isle of Man"
  },
  "Istanbul": {
    "exemplarCity": "Istanbul"
  },
  "Jersey": {
    "exemplarCity": "Jersey"
  },
  "Kaliningrad": {
    "exemplarCity": "Kaliningrad"
  },
  "Kiev": {
    "exemplarCity": "Kiew"
  },
  "Kirov": {
    "exemplarCity": "Kirow"
  },
  "Lisbon": {
    "exemplarCity": "Lissabon"
  },
  "Ljubljana": {
```

```
    "exemplarCity": "Ljubljana"
  },
  "London": {
    "long": {
      "daylight": "Britische Sommerzeit"
    },
    "exemplarCity": "London"
  },
  "Luxembourg": {
    "exemplarCity": "Luxemburg"
  },
  "Madrid": {
    "exemplarCity": "Madrid"
  },
  "Malta": {
    "exemplarCity": "Malta"
  },
  "Mariehamn": {
    "exemplarCity": "Mariehamn"
  },
  "Minsk": {
    "exemplarCity": "Minsk"
  },
  "Monaco": {
    "exemplarCity": "Monaco"
  },
  "Moscow": {
    "exemplarCity": "Moskau"
  },
  "Oslo": {
    "exemplarCity": "Oslo"
  },
  "Paris": {
    "exemplarCity": "Paris"
  },
  "Podgorica": {
    "exemplarCity": "Podgorica"
  },
  "Prague": {
    "exemplarCity": "Prag"
  },
  "Riga": {
    "exemplarCity": "Riga"
  },
  "Rome": {
    "exemplarCity": "Rom"
  },
  "Samara": {
    "exemplarCity": "Samara"
  },
  "San_Marino": {
    "exemplarCity": "San Marino"
  },
  "Sarajevo": {
    "exemplarCity": "Sarajevo"
  },
  "Simferopol": {
```

```
        "exemplarCity": "Simferopol"
      },
      "Skopje": {
        "exemplarCity": "Skopje"
      },
      "Sofia": {
        "exemplarCity": "Sofia"
      },
      "Stockholm": {
        "exemplarCity": "Stockholm"
      },
      "Tallinn": {
        "exemplarCity": "Tallinn"
      },
      "Tirane": {
        "exemplarCity": "Tirana"
      },
      "Ulyanovsk": {
        "exemplarCity": "Uljanowsk"
      },
      "Uzhgorod": {
        "exemplarCity": "Uschgorod"
      },
      "Vaduz": {
        "exemplarCity": "Vaduz"
      },
      "Vatican": {
        "exemplarCity": "Vatikan"
      },
      "Vienna": {
        "exemplarCity": "Wien"
      },
      "Vilnius": {
        "exemplarCity": "Vilnius"
      },
      "Volgograd": {
        "exemplarCity": "Wolgograd"
      },
      "Warsaw": {
        "exemplarCity": "Warschau"
      },
      "Zagreb": {
        "exemplarCity": "Zagreb"
      },
      "Zaporozhye": {
        "exemplarCity": "Saporischja"
      },
      "Zurich": {
        "exemplarCity": "Zürich"
      }
    },
    "Africa": {
      "Abidjan": {
        "exemplarCity": "Abidjan"
      },
      "Accra": {
        "exemplarCity": "Accra"
      }
    }
  }
}
```

```
    },
    "Addis_Ababa": {
      "exemplarCity": "Addis Abeba"
    },
    "Algiers": {
      "exemplarCity": "Algier"
    },
    "Asmera": {
      "exemplarCity": "Asmara"
    },
    "Bamako": {
      "exemplarCity": "Bamako"
    },
    "Bangui": {
      "exemplarCity": "Bangui"
    },
    "Banjul": {
      "exemplarCity": "Banjul"
    },
    "Bissau": {
      "exemplarCity": "Bissau"
    },
    "Blantyre": {
      "exemplarCity": "Blantyre"
    },
    "Brazzaville": {
      "exemplarCity": "Brazzaville"
    },
    "Bujumbura": {
      "exemplarCity": "Bujumbura"
    },
    "Cairo": {
      "exemplarCity": "Kairo"
    },
    "Casablanca": {
      "exemplarCity": "Casablanca"
    },
    "Ceuta": {
      "exemplarCity": "Ceuta"
    },
    "Conakry": {
      "exemplarCity": "Conakry"
    },
    "Dakar": {
      "exemplarCity": "Dakar"
    },
    "Dar_es_Salaam": {
      "exemplarCity": "Daressalam"
    },
    "Djibouti": {
      "exemplarCity": "Dschibuti"
    },
    "Douala": {
      "exemplarCity": "Douala"
    },
    "El_Aaiun": {
      "exemplarCity": "El Aaiún"
```

```
    },
    "Freetown": {
      "exemplarCity": "Freetown"
    },
    "Gaborone": {
      "exemplarCity": "Gaborone"
    },
    "Harare": {
      "exemplarCity": "Harare"
    },
    "Johannesburg": {
      "exemplarCity": "Johannesburg"
    },
    "Juba": {
      "exemplarCity": "Juba"
    },
    "Kampala": {
      "exemplarCity": "Kampala"
    },
    "Khartoum": {
      "exemplarCity": "Khartum"
    },
    "Kigali": {
      "exemplarCity": "Kigali"
    },
    "Kinshasa": {
      "exemplarCity": "Kinshasa"
    },
    "Lagos": {
      "exemplarCity": "Lagos"
    },
    "Libreville": {
      "exemplarCity": "Libreville"
    },
    "Lome": {
      "exemplarCity": "Lomé"
    },
    "Luanda": {
      "exemplarCity": "Luanda"
    },
    "Lubumbashi": {
      "exemplarCity": "Lubumbashi"
    },
    "Lusaka": {
      "exemplarCity": "Lusaka"
    },
    "Malabo": {
      "exemplarCity": "Malabo"
    },
    "Maputo": {
      "exemplarCity": "Maputo"
    },
    "Maseru": {
      "exemplarCity": "Maseru"
    },
    "Mbabane": {
      "exemplarCity": "Mbabane"
    }
  },
  "exemplarCity": "Mbabane"
}
```

```
    },
    "Mogadishu": {
      "exemplarCity": "Mogadischu"
    },
    "Monrovia": {
      "exemplarCity": "Monrovia"
    },
    "Nairobi": {
      "exemplarCity": "Nairobi"
    },
    "Ndjamena": {
      "exemplarCity": "N'Djamena"
    },
    "Niamey": {
      "exemplarCity": "Niamey"
    },
    "Nouakchott": {
      "exemplarCity": "Nouakchott"
    },
    "Ouagadougou": {
      "exemplarCity": "Ouagadougou"
    },
    "Porto-Novo": {
      "exemplarCity": "Porto Novo"
    },
    "Sao_Tome": {
      "exemplarCity": "São Tomé"
    },
    "Tripoli": {
      "exemplarCity": "Tripolis"
    },
    "Tunis": {
      "exemplarCity": "Tunis"
    },
    "Windhoek": {
      "exemplarCity": "Windhoek"
    }
  },
  "Asia": {
    "Aden": {
      "exemplarCity": "Aden"
    },
    "Almaty": {
      "exemplarCity": "Almaty"
    },
    "Amman": {
      "exemplarCity": "Amman"
    },
    "Anadyr": {
      "exemplarCity": "Anadyr"
    },
    "Aqtai": {
      "exemplarCity": "Aqtai"
    },
    "Aqtobe": {
      "exemplarCity": "Aktobe"
    }
  },
}
```



```
"Ashgabat": {
  "exemplarCity": "Aşgabat"
},
"Baghdad": {
  "exemplarCity": "Bagdad"
},
"Bahrain": {
  "exemplarCity": "Bahrain"
},
"Baku": {
  "exemplarCity": "Baku"
},
"Bangkok": {
  "exemplarCity": "Bangkok"
},
"Barnaul": {
  "exemplarCity": "Barnaul"
},
"Beirut": {
  "exemplarCity": "Beirut"
},
"Bishkek": {
  "exemplarCity": "Bischkek"
},
"Brunei": {
  "exemplarCity": "Brunei"
},
"Calcutta": {
  "exemplarCity": "Kalkutta"
},
"Chita": {
  "exemplarCity": "Tschita"
},
"Choibalsan": {
  "exemplarCity": "Tschoibalsan"
},
"Colombo": {
  "exemplarCity": "Colombo"
},
"Damascus": {
  "exemplarCity": "Damaskus"
},
"Dhaka": {
  "exemplarCity": "Dhaka"
},
"Dili": {
  "exemplarCity": "Dili"
},
"Dubai": {
  "exemplarCity": "Dubai"
},
"Dushanbe": {
  "exemplarCity": "Duschanbe"
},
"Gaza": {
  "exemplarCity": "Gaza"
},
```

```
"Hebron": {
  "exemplarCity": "Hebron"
},
"Hong_Kong": {
  "exemplarCity": "Hongkong"
},
"Hovd": {
  "exemplarCity": "Chowd"
},
"Irkutsk": {
  "exemplarCity": "Irkutsk"
},
"Jakarta": {
  "exemplarCity": "Jakarta"
},
"Jayapura": {
  "exemplarCity": "Jayapura"
},
"Jerusalem": {
  "exemplarCity": "Jerusalem"
},
"Kabul": {
  "exemplarCity": "Kabul"
},
"Kamchatka": {
  "exemplarCity": "Kamtschatka"
},
"Karachi": {
  "exemplarCity": "Karatschi"
},
"Katmandu": {
  "exemplarCity": "Kathmandu"
},
"Khandyga": {
  "exemplarCity": "Chandyga"
},
"Krasnoyarsk": {
  "exemplarCity": "Krasnojarsk"
},
"Kuala_Lumpur": {
  "exemplarCity": "Kuala Lumpur"
},
"Kuching": {
  "exemplarCity": "Kuching"
},
"Kuwait": {
  "exemplarCity": "Kuwait"
},
"Macau": {
  "exemplarCity": "Macao"
},
"Magadan": {
  "exemplarCity": "Magadan"
},
"Makassar": {
  "exemplarCity": "Makassar"
},
}
```

```
"Manila": {
  "exemplarCity": "Manila"
},
"Muscat": {
  "exemplarCity": "Maskat"
},
"Nicosia": {
  "exemplarCity": "Nikosia"
},
"Novokuznetsk": {
  "exemplarCity": "Nowokuznetsk"
},
"Novosibirsk": {
  "exemplarCity": "Nowosibirsk"
},
"Omsk": {
  "exemplarCity": "Omsk"
},
"Oral": {
  "exemplarCity": "Oral"
},
"Phnom_Penh": {
  "exemplarCity": "Phnom Penh"
},
"Pontianak": {
  "exemplarCity": "Pontianak"
},
"Pyongyang": {
  "exemplarCity": "Pjöngjang"
},
"Qatar": {
  "exemplarCity": "Katar"
},
"Qyzylorda": {
  "exemplarCity": "Qysylorda"
},
"Rangoon": {
  "exemplarCity": "Rangun"
},
"Riyadh": {
  "exemplarCity": "Riad"
},
"Saigon": {
  "exemplarCity": "Ho-Chi-Minh-Stadt"
},
"Sakhalin": {
  "exemplarCity": "Sachalin"
},
"Samarkand": {
  "exemplarCity": "Samarkand"
},
"Seoul": {
  "exemplarCity": "Seoul"
},
"Shanghai": {
  "exemplarCity": "Shanghai"
},
}
```

```
"Singapore": {
  "exemplarCity": "Singapur"
},
"Srednekolymsk": {
  "exemplarCity": "Srednekolymsk"
},
"Taipei": {
  "exemplarCity": "Taipeh"
},
"Tashkent": {
  "exemplarCity": "Taschkent"
},
"Tbilisi": {
  "exemplarCity": "Tiflis"
},
"Tehran": {
  "exemplarCity": "Teheran"
},
"Thimphu": {
  "exemplarCity": "Thimphu"
},
"Tokyo": {
  "exemplarCity": "Tokio"
},
"Tomsk": {
  "exemplarCity": "Tomsk"
},
"Ulaanbaatar": {
  "exemplarCity": "Ulaanbaatar"
},
"Urumqi": {
  "exemplarCity": "Ürümqi"
},
"Ust-Nera": {
  "exemplarCity": "Ust-Nera"
},
"Vientiane": {
  "exemplarCity": "Vientiane"
},
"Vladivostok": {
  "exemplarCity": "Wladiwostok"
},
"Yakutsk": {
  "exemplarCity": "Jakutsk"
},
"Yekaterinburg": {
  "exemplarCity": "Jekaterinburg"
},
"Yerevan": {
  "exemplarCity": "Eriwan"
}
},
"Indian": {
  "Antananarivo": {
    "exemplarCity": "Antananarivo"
  },
  "Chagos": {
```

```
        "exemplarCity": "Chagos"
      },
      "Christmas": {
        "exemplarCity": "Weihnachtsinsel"
      },
      "Cocos": {
        "exemplarCity": "Cocos"
      },
      "Comoro": {
        "exemplarCity": "Komoren"
      },
      "Kerguelen": {
        "exemplarCity": "Kerguelen"
      },
      "Mahe": {
        "exemplarCity": "Mahe"
      },
      "Maldives": {
        "exemplarCity": "Malediven"
      },
      "Mauritius": {
        "exemplarCity": "Mauritius"
      },
      "Mayotte": {
        "exemplarCity": "Mayotte"
      },
      "Reunion": {
        "exemplarCity": "Réunion"
      }
    },
    "Australia": {
      "Adelaide": {
        "exemplarCity": "Adelaide"
      },
      "Brisbane": {
        "exemplarCity": "Brisbane"
      },
      "Broken_Hill": {
        "exemplarCity": "Broken Hill"
      },
      "Currie": {
        "exemplarCity": "Currie"
      },
      "Darwin": {
        "exemplarCity": "Darwin"
      },
      "Eucla": {
        "exemplarCity": "Eucla"
      },
      "Hobart": {
        "exemplarCity": "Hobart"
      },
      "Lindeman": {
        "exemplarCity": "Lindeman"
      },
      "Lord_Howe": {
        "exemplarCity": "Lord Howe"
      }
    }
  }
}
```

```
    },  
    "Melbourne": {  
      "exemplarCity": "Melbourne"  
    },  
    "Perth": {  
      "exemplarCity": "Perth"  
    },  
    "Sydney": {  
      "exemplarCity": "Sydney"  
    }  
  },  
  "Pacific": {  
    "Apia": {  
      "exemplarCity": "Apia"  
    },  
    "Auckland": {  
      "exemplarCity": "Auckland"  
    },  
    "Bougainville": {  
      "exemplarCity": "Bougainville"  
    },  
    "Chatham": {  
      "exemplarCity": "Chatham"  
    },  
    "Easter": {  
      "exemplarCity": "Osterinsel"  
    },  
    "Efate": {  
      "exemplarCity": "Efate"  
    },  
    "Enderbury": {  
      "exemplarCity": "Enderbury"  
    },  
    "Fakaofu": {  
      "exemplarCity": "Fakaofu"  
    },  
    "Fiji": {  
      "exemplarCity": "Fidschi"  
    },  
    "Funafuti": {  
      "exemplarCity": "Funafuti"  
    },  
    "Galapagos": {  
      "exemplarCity": "Galapagos"  
    },  
    "Gambier": {  
      "exemplarCity": "Gambier"  
    },  
    "Guadalcanal": {  
      "exemplarCity": "Guadalcanal"  
    },  
    "Guam": {  
      "exemplarCity": "Guam"  
    },  
    "Honolulu": {  
      "exemplarCity": "Honolulu"  
    }  
  },  
}
```

```
"Johnston": {
  "exemplarCity": "Johnston"
},
"Kiritimati": {
  "exemplarCity": "Kiritimati"
},
"Kosrae": {
  "exemplarCity": "Kosrae"
},
"Kwajalein": {
  "exemplarCity": "Kwajalein"
},
"Majuro": {
  "exemplarCity": "Majuro"
},
"Marquesas": {
  "exemplarCity": "Marquesas"
},
"Midway": {
  "exemplarCity": "Midway"
},
"Nauru": {
  "exemplarCity": "Nauru"
},
"Niue": {
  "exemplarCity": "Niue"
},
"Norfolk": {
  "exemplarCity": "Norfolk"
},
"Noumea": {
  "exemplarCity": "Noumea"
},
"Pago_Pago": {
  "exemplarCity": "Pago Pago"
},
"Palau": {
  "exemplarCity": "Palau"
},
"Pitcairn": {
  "exemplarCity": "Pitcairn"
},
"Ponape": {
  "exemplarCity": "Pohnpei"
},
"Port_Moresby": {
  "exemplarCity": "Port Moresby"
},
"Rarotonga": {
  "exemplarCity": "Rarotonga"
},
"Saipan": {
  "exemplarCity": "Saipan"
},
"Tahiti": {
  "exemplarCity": "Tahiti"
},
}
```

```
"Tarawa": {
  "exemplarCity": "Tarawa"
},
"Tongatapu": {
  "exemplarCity": "Tongatapu"
},
"Truk": {
  "exemplarCity": "Chuuk"
},
"Wake": {
  "exemplarCity": "Wake"
},
"Wallis": {
  "exemplarCity": "Wallis"
}
},
"Arctic": {
  "Longyearbyen": {
    "exemplarCity": "Longyearbyen"
  }
},
"Antarctica": {
  "Casey": {
    "exemplarCity": "Casey"
  },
  "Davis": {
    "exemplarCity": "Davis"
  },
  "DumontDUrville": {
    "exemplarCity": "Dumont d'Urville"
  },
  "Macquarie": {
    "exemplarCity": "Macquarie"
  },
  "Mawson": {
    "exemplarCity": "Mawson"
  },
  "McMurdo": {
    "exemplarCity": "McMurdo"
  },
  "Palmer": {
    "exemplarCity": "Palmer"
  },
  "Rothera": {
    "exemplarCity": "Rothera"
  },
  "Syowa": {
    "exemplarCity": "Syowa"
  },
  "Troll": {
    "exemplarCity": "Troll"
  },
  "Vostok": {
    "exemplarCity": "Wostok"
  }
},
"Etc": {
```



```
"GMT": {
  "exemplarCity": "GMT"
},
"GMT1": {
  "exemplarCity": "GMT+1"
},
"GMT10": {
  "exemplarCity": "GMT+10"
},
"GMT11": {
  "exemplarCity": "GMT+11"
},
"GMT12": {
  "exemplarCity": "GMT+12"
},
"GMT2": {
  "exemplarCity": "GMT+2"
},
"GMT3": {
  "exemplarCity": "GMT+3"
},
"GMT4": {
  "exemplarCity": "GMT+4"
},
"GMT5": {
  "exemplarCity": "GMT+5"
},
"GMT6": {
  "exemplarCity": "GMT+6"
},
"GMT7": {
  "exemplarCity": "GMT+7"
},
"GMT8": {
  "exemplarCity": "GMT+8"
},
"GMT9": {
  "exemplarCity": "GMT+9"
},
"GMT-1": {
  "exemplarCity": "GMT-1"
},
"GMT-10": {
  "exemplarCity": "GMT-10"
},
"GMT-11": {
  "exemplarCity": "GMT-11"
},
"GMT-12": {
  "exemplarCity": "GMT-12"
},
"GMT-13": {
  "exemplarCity": "GMT-13"
},
"GMT-14": {
  "exemplarCity": "GMT-14"
},
},
```

```

    "GMT-2": {
      "exemplarCity": "GMT-2"
    },
    "GMT-3": {
      "exemplarCity": "GMT-3"
    },
    "GMT-4": {
      "exemplarCity": "GMT-4"
    },
    "GMT-5": {
      "exemplarCity": "GMT-5"
    },
    "GMT-6": {
      "exemplarCity": "GMT-6"
    },
    "GMT-7": {
      "exemplarCity": "GMT-7"
    },
    "GMT-8": {
      "exemplarCity": "GMT-8"
    },
    "GMT-9": {
      "exemplarCity": "GMT-9"
    },
    "Unknown": {
      "exemplarCity": "Unbekannt"
    }
  },
  "metazone": {
    "Acre": {
      "long": {
        "generic": "Acre-Zeit",
        "standard": "Acre-Normalzeit",
        "daylight": "Acre-Sommerzeit"
      }
    },
    "Afghanistan": {
      "long": {
        "standard": "Afghanistan-Zeit"
      }
    },
    "Africa_Central": {
      "long": {
        "standard": "Zentralafrikanische Zeit"
      }
    },
    "Africa_Eastern": {
      "long": {
        "standard": "Ostafrikanische Zeit"
      }
    },
    "Africa_Southern": {
      "long": {
        "standard": "Südafrikanische Zeit"
      }
    }
  },

```

```

"Africa_Western": {
  "long": {
    "generic": "Westafrikanische Zeit",
    "standard": "Westafrikanische Normalzeit",
    "daylight": "Westafrikanische Sommerzeit"
  }
},
"Alaska": {
  "long": {
    "generic": "Alaska-Zeit",
    "standard": "Alaska-Normalzeit",
    "daylight": "Alaska-Sommerzeit"
  }
},
"Almaty": {
  "long": {
    "generic": "Almaty-Zeit",
    "standard": "Almaty-Normalzeit",
    "daylight": "Almaty-Sommerzeit"
  }
},
"Amazon": {
  "long": {
    "generic": "Amazonas-Zeit",
    "standard": "Amazonas-Normalzeit",
    "daylight": "Amazonas-Sommerzeit"
  }
},
"America_Central": {
  "long": {
    "generic": "Nordamerikanische Inlandzeit",
    "standard": "Nordamerikanische Inland-Normalzeit",
    "daylight": "Nordamerikanische Inland-Sommerzeit"
  }
},
"America_Eastern": {
  "long": {
    "generic": "Nordamerikanische Ostküstenzeit",
    "standard": "Nordamerikanische Ostküsten-Normalzeit",
    "daylight": "Nordamerikanische Ostküsten-Sommerzeit"
  }
},
"America_Mountain": {
  "long": {
    "generic": "Rocky-Mountain-Zeit",
    "standard": "Rocky Mountain-Normalzeit",
    "daylight": "Rocky-Mountain-Sommerzeit"
  }
},
"America_Pacific": {
  "long": {
    "generic": "Nordamerikanische Westküstenzeit",
    "standard": "Nordamerikanische Westküsten-Normalzeit",
    "daylight": "Nordamerikanische Westküsten-Sommerzeit"
  }
},
"Anadyr": {

```

```

        "long": {
            "generic": "Anadyr Zeit",
            "standard": "Anadyr Normalzeit",
            "daylight": "Anadyr Sommerzeit"
        }
    },
    "Apia": {
        "long": {
            "generic": "Apia-Zeit",
            "standard": "Apia-Normalzeit",
            "daylight": "Apia-Sommerzeit"
        }
    },
    "Aqttau": {
        "long": {
            "generic": "Aqttau-Zeit",
            "standard": "Aqttau-Normalzeit",
            "daylight": "Aqttau-Sommerzeit"
        }
    },
    "Aqtobe": {
        "long": {
            "generic": "Aqtöbe-Zeit",
            "standard": "Aqtöbe-Normalzeit",
            "daylight": "Aqtöbe-Sommerzeit"
        }
    },
    "Arabian": {
        "long": {
            "generic": "Arabische Zeit",
            "standard": "Arabische Normalzeit",
            "daylight": "Arabische Sommerzeit"
        }
    },
    "Argentina": {
        "long": {
            "generic": "Argentinische Zeit",
            "standard": "Argentinische Normalzeit",
            "daylight": "Argentinische Sommerzeit"
        }
    },
    "Argentina_Western": {
        "long": {
            "generic": "Westargentinische Zeit",
            "standard": "Westargentinische Normalzeit",
            "daylight": "Westargentinische Sommerzeit"
        }
    },
    "Armenia": {
        "long": {
            "generic": "Armenische Zeit",
            "standard": "Armenische Normalzeit",
            "daylight": "Armenische Sommerzeit"
        }
    },
    "Atlantic": {
        "long": {

```

```

        "generic": "Atlantik-Zeit",
        "standard": "Atlantik-Normalzeit",
        "daylight": "Atlantik-Sommerzeit"
    }
},
"Australia_Central": {
    "long": {
        "generic": "Zentralaustralische Zeit",
        "standard": "Zentralaustralische Normalzeit",
        "daylight": "Zentralaustralische Sommerzeit"
    }
},
"Australia_CentralWestern": {
    "long": {
        "generic": "Zentral-/Westaustralische Zeit",
        "standard": "Zentral-/Westaustralische Normalzeit",
        "daylight": "Zentral-/Westaustralische Sommerzeit"
    }
},
"Australia_Eastern": {
    "long": {
        "generic": "Ostaustralische Zeit",
        "standard": "Ostaustralische Normalzeit",
        "daylight": "Ostaustralische Sommerzeit"
    }
},
"Australia_Western": {
    "long": {
        "generic": "Westaustralische Zeit",
        "standard": "Westaustralische Normalzeit",
        "daylight": "Westaustralische Sommerzeit"
    }
},
"Azerbaijan": {
    "long": {
        "generic": "Aserbaidsschanische Zeit",
        "standard": "Aserbeidschanische Normalzeit",
        "daylight": "Aserbaidsschanische Sommerzeit"
    }
},
"Azores": {
    "long": {
        "generic": "Azoren-Zeit",
        "standard": "Azoren-Normalzeit",
        "daylight": "Azoren-Sommerzeit"
    }
},
"Bangladesh": {
    "long": {
        "generic": "Bangladesch-Zeit",
        "standard": "Bangladesch-Normalzeit",
        "daylight": "Bangladesch-Sommerzeit"
    }
},
"Bhutan": {
    "long": {
        "standard": "Bhutan-Zeit"
    }
}

```

```

    }
  },
  "Bolivia": {
    "long": {
      "standard": "Bolivianische Zeit"
    }
  },
  "Brasilia": {
    "long": {
      "generic": "Brasília-Zeit",
      "standard": "Brasília-Normalzeit",
      "daylight": "Brasília-Sommerzeit"
    }
  },
  "Brunei": {
    "long": {
      "standard": "Brunei-Zeit"
    }
  },
  "Cape_Verde": {
    "long": {
      "generic": "Cabo-Verde-Zeit",
      "standard": "Cabo-Verde-Normalzeit",
      "daylight": "Cabo-Verde-Sommerzeit"
    }
  },
  "Casey": {
    "long": {
      "standard": "Casey-Zeit"
    }
  },
  "Chamorro": {
    "long": {
      "standard": "Chamorro-Zeit"
    }
  },
  "Chatham": {
    "long": {
      "generic": "Chatham-Zeit",
      "standard": "Chatham-Normalzeit",
      "daylight": "Chatham-Sommerzeit"
    }
  },
  "Chile": {
    "long": {
      "generic": "Chilenische Zeit",
      "standard": "Chilenische Normalzeit",
      "daylight": "Chilenische Sommerzeit"
    }
  },
  "China": {
    "long": {
      "generic": "Chinesische Zeit",
      "standard": "Chinesische Normalzeit",
      "daylight": "Chinesische Sommerzeit"
    }
  },
  },

```

```

"Choibalsan": {
  "long": {
    "generic": "Tschoibalsan-Zeit",
    "standard": "Tschoibalsan-Normalzeit",
    "daylight": "Tschoibalsan-Sommerzeit"
  }
},
"Christmas": {
  "long": {
    "standard": "Weihnachtsinsel-Zeit"
  }
},
"Cocos": {
  "long": {
    "standard": "Kokosinseln-Zeit"
  }
},
"Colombia": {
  "long": {
    "generic": "Kolumbianische Zeit",
    "standard": "Kolumbianische Normalzeit",
    "daylight": "Kolumbianische Sommerzeit"
  }
},
"Cook": {
  "long": {
    "generic": "Cookinseln-Zeit",
    "standard": "Cookinseln-Normalzeit",
    "daylight": "Cookinseln-Sommerzeit"
  }
},
"Cuba": {
  "long": {
    "generic": "Kubanische Zeit",
    "standard": "Kubanische Normalzeit",
    "daylight": "Kubanische Sommerzeit"
  }
},
"Davis": {
  "long": {
    "standard": "Davis-Zeit"
  }
},
"DumontDUrville": {
  "long": {
    "standard": "Dumont-d'Urville-Zeit"
  }
},
"East_Timor": {
  "long": {
    "standard": "Osttimor-Zeit"
  }
},
"Easter": {
  "long": {
    "generic": "Osterinsel-Zeit",
    "standard": "Osterinsel-Normalzeit",

```

```

        "daylight": "Osterinsel-Sommerzeit"
    },
    "Ecuador": {
        "long": {
            "standard": "Ecuadorianische Zeit"
        }
    },
    "Europe_Central": {
        "long": {
            "generic": "Mitteleuropäische Zeit",
            "standard": "Mitteleuropäische Normalzeit",
            "daylight": "Mitteleuropäische Sommerzeit"
        },
        "short": {
            "generic": "MEZ",
            "standard": "MEZ",
            "daylight": "MESZ"
        }
    },
    "Europe_Eastern": {
        "long": {
            "generic": "Osteuropäische Zeit",
            "standard": "Osteuropäische Normalzeit",
            "daylight": "Osteuropäische Sommerzeit"
        },
        "short": {
            "generic": "OEZ",
            "standard": "OEZ",
            "daylight": "OESZ"
        }
    },
    "Europe_Further_Eastern": {
        "long": {
            "standard": "Kaliningrader Zeit"
        }
    },
    "Europe_Western": {
        "long": {
            "generic": "Westeuropäische Zeit",
            "standard": "Westeuropäische Normalzeit",
            "daylight": "Westeuropäische Sommerzeit"
        },
        "short": {
            "generic": "WEZ",
            "standard": "WEZ",
            "daylight": "WESZ"
        }
    },
    "Falkland": {
        "long": {
            "generic": "Falklandinseln-Zeit",
            "standard": "Falklandinseln-Normalzeit",
            "daylight": "Falklandinseln-Sommerzeit"
        }
    },
    "Fiji": {

```



```

        "long": {
            "generic": "Fidschi-Zeit",
            "standard": "Fidschi-Normalzeit",
            "daylight": "Fidschi-Sommerzeit"
        }
    },
    "French_Guiana": {
        "long": {
            "standard": "Französisch-Guayana-Zeit"
        }
    },
    "French_Southern": {
        "long": {
            "standard": "Französische Süd- und Antarktisgebiete-Zeit"
        }
    },
    "Galapagos": {
        "long": {
            "standard": "Galapagos-Zeit"
        }
    },
    "Gambier": {
        "long": {
            "standard": "Gambier-Zeit"
        }
    },
    "Georgia": {
        "long": {
            "generic": "Georgische Zeit",
            "standard": "Georgische Normalzeit",
            "daylight": "Georgische Sommerzeit"
        }
    },
    "Gilbert_Islands": {
        "long": {
            "standard": "Gilbert-Inseln-Zeit"
        }
    },
    "GMT": {
        "long": {
            "standard": "Mittlere Greenwich-Zeit"
        }
    },
    "Greenland_Eastern": {
        "long": {
            "generic": "Ostgrönland-Zeit",
            "standard": "Ostgrönland-Normalzeit",
            "daylight": "Ostgrönland-Sommerzeit"
        }
    },
    "Greenland_Western": {
        "long": {
            "generic": "Westgrönland-Zeit",
            "standard": "Westgrönland-Normalzeit",
            "daylight": "Westgrönland-Sommerzeit"
        }
    },
    },

```

```

"Guam": {
  "long": {
    "standard": "Guam-Zeit"
  }
},
"Gulf": {
  "long": {
    "standard": "Golf-Zeit"
  }
},
"Guyana": {
  "long": {
    "standard": "Guyana-Zeit"
  }
},
"Hawaii_Aleutian": {
  "long": {
    "generic": "Hawaii-Aleuten-Zeit",
    "standard": "Hawaii-Aleuten-Normalzeit",
    "daylight": "Hawaii-Aleuten-Sommerzeit"
  }
},
"Hong_Kong": {
  "long": {
    "generic": "Hongkong-Zeit",
    "standard": "Hongkong-Normalzeit",
    "daylight": "Hongkong-Sommerzeit"
  }
},
"Hovd": {
  "long": {
    "generic": "Chowd-Zeit",
    "standard": "Chowd-Normalzeit",
    "daylight": "Chowd-Sommerzeit"
  }
},
"India": {
  "long": {
    "standard": "Indische Zeit"
  }
},
"Indian_Ocean": {
  "long": {
    "standard": "Indischer Ozean-Zeit"
  }
},
"Indochina": {
  "long": {
    "standard": "Indochina-Zeit"
  }
},
"Indonesia_Central": {
  "long": {
    "standard": "Zentralindonesische Zeit"
  }
},
"Indonesia_Eastern": {

```

```

        "long": {
            "standard": "Ostindonesische Zeit"
        }
    },
    "Indonesia_Western": {
        "long": {
            "standard": "Westindonesische Zeit"
        }
    },
    "Iran": {
        "long": {
            "generic": "Iranische Zeit",
            "standard": "Iranische Normalzeit",
            "daylight": "Iranische Sommerzeit"
        }
    },
    "Irkutsk": {
        "long": {
            "generic": "Irkutsk-Zeit",
            "standard": "Irkutsk-Normalzeit",
            "daylight": "Irkutsk-Sommerzeit"
        }
    },
    "Israel": {
        "long": {
            "generic": "Israelische Zeit",
            "standard": "Israelische Normalzeit",
            "daylight": "Israelische Sommerzeit"
        }
    },
    "Japan": {
        "long": {
            "generic": "Japanische Zeit",
            "standard": "Japanische Normalzeit",
            "daylight": "Japanische Sommerzeit"
        }
    },
    "Kamchatka": {
        "long": {
            "generic": "Kamtschatka-Zeit",
            "standard": "Kamtschatka-Normalzeit",
            "daylight": "Kamtschatka-Sommerzeit"
        }
    },
    "Kazakhstan_Eastern": {
        "long": {
            "standard": "Ostkasachische Zeit"
        }
    },
    "Kazakhstan_Western": {
        "long": {
            "standard": "Westkasachische Zeit"
        }
    },
    "Korea": {
        "long": {
            "generic": "Koreanische Zeit",

```

```

        "standard": "Koreanische Normalzeit",
        "daylight": "Koreanische Sommerzeit"
    },
    "Kosrae": {
        "long": {
            "standard": "Kosrae-Zeit"
        }
    },
    "Krasnoyarsk": {
        "long": {
            "generic": "Krasnojarsk-Zeit",
            "standard": "Krasnojarsk-Normalzeit",
            "daylight": "Krasnojarsk-Sommerzeit"
        }
    },
    "Kyrgystan": {
        "long": {
            "standard": "Kirgisistan-Zeit"
        }
    },
    "Lanka": {
        "long": {
            "standard": "Sri-Lanka-Zeit"
        }
    },
    "Line_Islands": {
        "long": {
            "standard": "Linieninseln-Zeit"
        }
    },
    "Lord_Howe": {
        "long": {
            "generic": "Lord-Howe-Zeit",
            "standard": "Lord-Howe-Normalzeit",
            "daylight": "Lord-Howe-Sommerzeit"
        }
    },
    "Macau": {
        "long": {
            "generic": "Macau-Zeit",
            "standard": "Macau-Normalzeit",
            "daylight": "Macau-Sommerzeit"
        }
    },
    "Macquarie": {
        "long": {
            "standard": "Macquarieinsel-Zeit"
        }
    },
    "Magadan": {
        "long": {
            "generic": "Magadan-Zeit",
            "standard": "Magadan-Normalzeit",
            "daylight": "Magadan-Sommerzeit"
        }
    },
    },

```

```
"Malaysia": {
  "long": {
    "standard": "Malaysische Zeit"
  }
},
"Maldives": {
  "long": {
    "standard": "Malediven-Zeit"
  }
},
"Marquesas": {
  "long": {
    "standard": "Marquesas-Zeit"
  }
},
"Marshall_Islands": {
  "long": {
    "standard": "Marshallinseln-Zeit"
  }
},
"Mauritius": {
  "long": {
    "generic": "Mauritius-Zeit",
    "standard": "Mauritius-Normalzeit",
    "daylight": "Mauritius-Sommerzeit"
  }
},
"Mawson": {
  "long": {
    "standard": "Mawson-Zeit"
  }
},
"Mexico_Northwest": {
  "long": {
    "generic": "Mexiko Nordwestliche Zone-Zeit",
    "standard": "Mexiko Nordwestliche Zone-Normalzeit",
    "daylight": "Mexiko Nordwestliche Zone-Sommerzeit"
  }
},
"Mexico_Pacific": {
  "long": {
    "generic": "Mexiko Pazifikzone-Zeit",
    "standard": "Mexiko Pazifikzone-Normalzeit",
    "daylight": "Mexiko Pazifikzone-Sommerzeit"
  }
},
"Mongolia": {
  "long": {
    "generic": "Ulaanbaatar-Zeit",
    "standard": "Ulaanbaatar-Normalzeit",
    "daylight": "Ulaanbaatar-Sommerzeit"
  }
},
"Moscow": {
  "long": {
    "generic": "Moskauer Zeit",
    "standard": "Moskauer Normalzeit",
```

```

        "daylight": "Moskauer Sommerzeit"
    }
},
"Myanmar": {
    "long": {
        "standard": "Myanmar-Zeit"
    }
},
"Nauru": {
    "long": {
        "standard": "Nauru-Zeit"
    }
},
"Nepal": {
    "long": {
        "standard": "Nepalesische Zeit"
    }
},
"New_Caledonia": {
    "long": {
        "generic": "Neukaledonische Zeit",
        "standard": "Neukaledonische Normalzeit",
        "daylight": "Neukaledonische Sommerzeit"
    }
},
"New_Zealand": {
    "long": {
        "generic": "Neuseeland-Zeit",
        "standard": "Neuseeland-Normalzeit",
        "daylight": "Neuseeland-Sommerzeit"
    }
},
"Newfoundland": {
    "long": {
        "generic": "Neufundland-Zeit",
        "standard": "Neufundland-Normalzeit",
        "daylight": "Neufundland-Sommerzeit"
    }
},
"Niue": {
    "long": {
        "standard": "Niue-Zeit"
    }
},
"Norfolk": {
    "long": {
        "standard": "Norfolkinsel-Zeit"
    }
},
"Noronha": {
    "long": {
        "generic": "Fernando de Noronha-Zeit",
        "standard": "Fernando de Noronha-Normalzeit",
        "daylight": "Fernando de Noronha-Sommerzeit"
    }
},
"North_Mariana": {

```

```

        "long": {
            "standard": "Nördliche-Marianen-Zeit"
        }
    },
    "Novosibirsk": {
        "long": {
            "generic": "Nowosibirsk-Zeit",
            "standard": "Nowosibirsk-Normalzeit",
            "daylight": "Nowosibirsk-Sommerzeit"
        }
    },
    "Omsk": {
        "long": {
            "generic": "Omsk-Zeit",
            "standard": "Omsk-Normalzeit",
            "daylight": "Omsk-Sommerzeit"
        }
    },
    "Pakistan": {
        "long": {
            "generic": "Pakistanische Zeit",
            "standard": "Pakistanische Normalzeit",
            "daylight": "Pakistanische Sommerzeit"
        }
    },
    "Palau": {
        "long": {
            "standard": "Palau-Zeit"
        }
    },
    "Papua_New_Guinea": {
        "long": {
            "standard": "Papua-Neuguinea-Zeit"
        }
    },
    "Paraguay": {
        "long": {
            "generic": "Paraguayische Zeit",
            "standard": "Paraguayische Normalzeit",
            "daylight": "Paraguayische Sommerzeit"
        }
    },
    "Peru": {
        "long": {
            "generic": "Peruanische Zeit",
            "standard": "Peruanische Normalzeit",
            "daylight": "Peruanische Sommerzeit"
        }
    },
    "Philippines": {
        "long": {
            "generic": "Philippinische Zeit",
            "standard": "Philippinische Normalzeit",
            "daylight": "Philippinische Sommerzeit"
        }
    },
    "Phoenix_Islands": {

```

```

        "long": {
            "standard": "Phoenixinseln-Zeit"
        }
    },
    "Pierre_Miquelon": {
        "long": {
            "generic": "Saint-Pierre-und-Miquelon-Zeit",
            "standard": "Saint-Pierre-und-Miquelon-Normalzeit",
            "daylight": "Saint-Pierre-und-Miquelon-Sommerzeit"
        }
    },
    "Pitcairn": {
        "long": {
            "standard": "Pitcairninseln-Zeit"
        }
    },
    "Ponape": {
        "long": {
            "standard": "Ponape-Zeit"
        }
    },
    "Pyongyang": {
        "long": {
            "standard": "Pjöngjang-Zeit"
        }
    },
    "Qyzylorda": {
        "long": {
            "generic": "Quysylorda-Zeit",
            "standard": "Quysylorda-Normalzeit",
            "daylight": "Qysylorda-Sommerzeit"
        }
    },
    "Reunion": {
        "long": {
            "standard": "Réunion-Zeit"
        }
    },
    "Rothera": {
        "long": {
            "standard": "Rothera-Zeit"
        }
    },
    "Sakhalin": {
        "long": {
            "generic": "Sachalin-Zeit",
            "standard": "Sachalin-Normalzeit",
            "daylight": "Sachalin-Sommerzeit"
        }
    },
    "Samara": {
        "long": {
            "generic": "Samara-Zeit",
            "standard": "Samara-Normalzeit",
            "daylight": "Samara-Sommerzeit"
        }
    },
    },

```



```

"Samoa": {
  "long": {
    "generic": "Samoa-Zeit",
    "standard": "Samoa-Normalzeit",
    "daylight": "Samoa-Sommerzeit"
  }
},
"Seychelles": {
  "long": {
    "standard": "Seychellen-Zeit"
  }
},
"Singapore": {
  "long": {
    "standard": "Singapur-Zeit"
  }
},
"Solomon": {
  "long": {
    "standard": "Salomoninseln-Zeit"
  }
},
"South_Georgia": {
  "long": {
    "standard": "Südgeorgische Zeit"
  }
},
"Suriname": {
  "long": {
    "standard": "Suriname-Zeit"
  }
},
"Syowa": {
  "long": {
    "standard": "Syowa-Zeit"
  }
},
"Tahiti": {
  "long": {
    "standard": "Tahiti-Zeit"
  }
},
"Taipei": {
  "long": {
    "generic": "Taipeh-Zeit",
    "standard": "Taipeh-Normalzeit",
    "daylight": "Taipeh-Sommerzeit"
  }
},
"Tajikistan": {
  "long": {
    "standard": "Tadschikistan-Zeit"
  }
},
"Tokelau": {
  "long": {
    "standard": "Tokelau-Zeit"
  }
}

```

```

    }
  },
  "Tonga": {
    "long": {
      "generic": "Tonganische Zeit",
      "standard": "Tonganische Normalzeit",
      "daylight": "Tonganische Sommerzeit"
    }
  },
  "Truk": {
    "long": {
      "standard": "Chuuk-Zeit"
    }
  },
  "Turkmenistan": {
    "long": {
      "generic": "Turkmenistan-Zeit",
      "standard": "Turkmenistan-Normalzeit",
      "daylight": "Turkmenistan-Sommerzeit"
    }
  },
  "Tuvalu": {
    "long": {
      "standard": "Tuvalu-Zeit"
    }
  },
  "Uruguay": {
    "long": {
      "generic": "Uruguayische Zeit",
      "standard": "Uruguayische Normalzeit",
      "daylight": "Uruguayische Sommerzeit"
    }
  },
  "Uzbekistan": {
    "long": {
      "generic": "Usbekistan-Zeit",
      "standard": "Usbekistan-Normalzeit",
      "daylight": "Usbekistan-Sommerzeit"
    }
  },
  "Vanuatu": {
    "long": {
      "generic": "Vanuatu-Zeit",
      "standard": "Vanuatu-Normalzeit",
      "daylight": "Vanuatu-Sommerzeit"
    }
  },
  "Venezuela": {
    "long": {
      "standard": "Venezuela-Zeit"
    }
  },
  "Vladivostok": {
    "long": {
      "generic": "Wladiwostok-Zeit",
      "standard": "Wladiwostok-Normalzeit",
      "daylight": "Wladiwostok-Sommerzeit"
    }
  }
}

```

```

    }
  },
  "Volgograd": {
    "long": {
      "generic": "Wolgograd-Zeit",
      "standard": "Wolgograd-Normalzeit",
      "daylight": "Wolgograd-Sommerzeit"
    }
  },
  "Vostok": {
    "long": {
      "standard": "Wostok-Zeit"
    }
  },
  "Wake": {
    "long": {
      "standard": "Wake-Insel-Zeit"
    }
  },
  "Wallis": {
    "long": {
      "standard": "Wallis-und-Futuna-Zeit"
    }
  },
  "Yakutsk": {
    "long": {
      "generic": "Jakutsk-Zeit",
      "standard": "Jakutsk-Normalzeit",
      "daylight": "Jakutsk-Sommerzeit"
    }
  },
  "Yekaterinburg": {
    "long": {
      "generic": "Jekaterinburg-Zeit",
      "standard": "Jekaterinburg-Normalzeit",
      "daylight": "Jekaterinburg-Sommerzeit"
    }
  }
}

```

TIMEZONENAMES.JSX

```

{
  "main";
  {
    "de";
    {
      "identity";
      {
        "version";
        {

```

```

        "_number";
        "$Revision: 12879 $",
        "_cldrVersion";
        "30.0.3";
    }
    "language";
    "de";
}
"dates";
{
    "timeZoneNames";
    {
        "hourFormat";
        "+HH:mm;-HH:mm",
        "gmtFormat";
        "GMT{0}",
        "gmtZeroFormat";
        "GMT",
        "regionFormat";
        "{0} Zeit",
        "regionFormat-type-daylight";
        "{0} Sommerzeit",
        "regionFormat-type-standard";
        "{0} Normalzeit",
        "fallbackFormat";
        "{1} ({0})",
        "zone";
        {
            "America";
            {
                "Adak";
                {
                    "exemplarCity";
                    "Adak";
                }
                "Anchorage";
                {
                    "exemplarCity";
                    "Anchorage";
                }
                "Anguilla";
                {
                    "exemplarCity";
                    "Anguilla";
                }
                "Antigua";
                {
                    "exemplarCity";
                    "Antigua";
                }
                "Araguaina";
                {
                    "exemplarCity";
                    "Araguaina";
                }
                "Argentina";
                {

```

```
"Rio_Gallegos";
{
  "exemplarCity";
  "Rio Gallegos";
}
"San_Juan";
{
  "exemplarCity";
  "San Juan";
}
"Ushuaia";
{
  "exemplarCity";
  "Ushuaia";
}
"La_Rioja";
{
  "exemplarCity";
  "La Rioja";
}
"San_Luis";
{
  "exemplarCity";
  "San Luis";
}
"Salta";
{
  "exemplarCity";
  "Salta";
}
"Tucuman";
{
  "exemplarCity";
  "Tucuman";
}
}
"Aruba";
{
  "exemplarCity";
  "Aruba";
}
"Asuncion";
{
  "exemplarCity";
  "Asunción";
}
"Bahia";
{
  "exemplarCity";
  "Bahia";
}
"Bahia_Banderas";
{
  "exemplarCity";
  "Bahia Banderas";
}
"Barbados";
```

```
{
    "exemplarCity";
    "Barbados";
}
"Belem";
{
    "exemplarCity";
    "Belem";
}
"Belize";
{
    "exemplarCity";
    "Belize";
}
"Blanc-Sablon";
{
    "exemplarCity";
    "Blanc-Sablon";
}
"Boa_Vista";
{
    "exemplarCity";
    "Boa Vista";
}
"Bogota";
{
    "exemplarCity";
    "Bogotá";
}
"Boise";
{
    "exemplarCity";
    "Boise";
}
"Buenos_Aires";
{
    "exemplarCity";
    "Buenos Aires";
}
"Cambridge_Bay";
{
    "exemplarCity";
    "Cambridge Bay";
}
"Campo_Grande";
{
    "exemplarCity";
    "Campo Grande";
}
"Cancun";
{
    "exemplarCity";
    "Cancún";
}
"Caracas";
{
    "exemplarCity";
```

```
        "Caracas";
    }
    "Catamarca";
    {
        "exemplarCity";
        "Catamarca";
    }
    "Cayenne";
    {
        "exemplarCity";
        "Cayenne";
    }
    "Cayman";
    {
        "exemplarCity";
        "Kaimaninseln";
    }
    "Chicago";
    {
        "exemplarCity";
        "Chicago";
    }
    "Chihuahua";
    {
        "exemplarCity";
        "Chihuahua";
    }
    "Coral_Harbour";
    {
        "exemplarCity";
        "Atikokan";
    }
    "Cordoba";
    {
        "exemplarCity";
        "Córdoba";
    }
    "Costa_Rica";
    {
        "exemplarCity";
        "Costa Rica";
    }
    "Creston";
    {
        "exemplarCity";
        "Creston";
    }
    "Cuiaba";
    {
        "exemplarCity";
        "Cuiaba";
    }
    "Curacao";
    {
        "exemplarCity";
        "Curaçao";
    }
}
```

```
"Danmarkshavn";
{
  "exemplarCity";
  "Danmarkshavn";
}
"Dawson";
{
  "exemplarCity";
  "Dawson";
}
"Dawson_Creek";
{
  "exemplarCity";
  "Dawson Creek";
}
"Denver";
{
  "exemplarCity";
  "Denver";
}
"Detroit";
{
  "exemplarCity";
  "Detroit";
}
"Dominica";
{
  "exemplarCity";
  "Dominica";
}
"Edmonton";
{
  "exemplarCity";
  "Edmonton";
}
"Eirunepe";
{
  "exemplarCity";
  "Eirunepe";
}
"El_Salvador";
{
  "exemplarCity";
  "El Salvador";
}
"Fort_Nelson";
{
  "exemplarCity";
  "Fort Nelson";
}
"Fortaleza";
{
  "exemplarCity";
  "Fortaleza";
}
"Glace_Bay";
{
```



```
        "exemplarCity";
        "Glace Bay";
    }
    "Godthab";
    {
        "exemplarCity";
        "Nuuk";
    }
    "Goose_Bay";
    {
        "exemplarCity";
        "Goose Bay";
    }
    "Grand_Turk";
    {
        "exemplarCity";
        "Grand Turk";
    }
    "Grenada";
    {
        "exemplarCity";
        "Grenada";
    }
    "Guadeloupe";
    {
        "exemplarCity";
        "Guadeloupe";
    }
    "Guatemala";
    {
        "exemplarCity";
        "Guatemala";
    }
    "Guayaquil";
    {
        "exemplarCity";
        "Guayaquil";
    }
    "Guyana";
    {
        "exemplarCity";
        "Guyana";
    }
    "Halifax";
    {
        "exemplarCity";
        "Halifax";
    }
    "Havana";
    {
        "exemplarCity";
        "Havanna";
    }
    "Hermosillo";
    {
        "exemplarCity";
        "Hermosillo";
    }
}
```

```

    }
    "Indiana";
    {
        "Vincennes";
        {
            "exemplarCity";
            "Vincennes, Indiana";
        }
        "Petersburg";
        {
            "exemplarCity";
            "Petersburg, Indiana";
        }
        "Tell_City";
        {
            "exemplarCity";
            "Tell City, Indiana";
        }
        "Knox";
        {
            "exemplarCity";
            "Knox, Indiana";
        }
        "Winamac";
        {
            "exemplarCity";
            "Winamac, Indiana";
        }
        "Marengo";
        {
            "exemplarCity";
            "Marengo, Indiana";
        }
        "Vevay";
        {
            "exemplarCity";
            "Vevay, Indiana";
        }
    }
    "Indianapolis";
    {
        "exemplarCity";
        "Indianapolis";
    }
    "Inuvik";
    {
        "exemplarCity";
        "Inuvik";
    }
    "Iqaluit";
    {
        "exemplarCity";
        "Iqaluit";
    }
    "Jamaica";
    {
        "exemplarCity";
    }

```

```

        "Jamaika";
    }
    "Jujuy";
    {
        "exemplarCity";
        "Jujuy";
    }
    "Juneau";
    {
        "exemplarCity";
        "Juneau";
    }
    "Kentucky";
    {
        "Monticello";
        {
            "exemplarCity";
            "Monticello, Kentucky";
        }
    }
    "Kralendijk";
    {
        "exemplarCity";
        "Kralendijk";
    }
    "La_Paz";
    {
        "exemplarCity";
        "La Paz";
    }
    "Lima";
    {
        "exemplarCity";
        "Lima";
    }
    "Los_Angeles";
    {
        "exemplarCity";
        "Los Angeles";
    }
    "Louisville";
    {
        "exemplarCity";
        "Louisville";
    }
    "Lower_Princes";
    {
        "exemplarCity";
        "Lower Prince's Quarter";
    }
    "Maceio";
    {
        "exemplarCity";
        "Maceio";
    }
    "Managua";
    {

```

```
        "exemplarCity";
        "Managua";
    }
    "Manaus";
    {
        "exemplarCity";
        "Manaus";
    }
    "Marigot";
    {
        "exemplarCity";
        "Marigot";
    }
    "Martinique";
    {
        "exemplarCity";
        "Martinique";
    }
    "Matamoros";
    {
        "exemplarCity";
        "Matamoros";
    }
    "Mazatlan";
    {
        "exemplarCity";
        "Mazatlan";
    }
    "Mendoza";
    {
        "exemplarCity";
        "Mendoza";
    }
    "Menominee";
    {
        "exemplarCity";
        "Menominee";
    }
    "Merida";
    {
        "exemplarCity";
        "Merida";
    }
    "Metlakatla";
    {
        "exemplarCity";
        "Metlakatla";
    }
    "Mexico_City";
    {
        "exemplarCity";
        "Mexiko-Stadt";
    }
    "Miquelon";
    {
        "exemplarCity";
        "Miquelon";
    }
```

```
}
"Moncton";
{
  "exemplarCity";
  "Moncton";
}
"Monterrey";
{
  "exemplarCity";
  "Monterrey";
}
"Montevideo";
{
  "exemplarCity";
  "Montevideo";
}
"Montserrat";
{
  "exemplarCity";
  "Montserrat";
}
"Nassau";
{
  "exemplarCity";
  "Nassau";
}
"New_York";
{
  "exemplarCity";
  "New York";
}
"Nipigon";
{
  "exemplarCity";
  "Nipigon";
}
"Nome";
{
  "exemplarCity";
  "Nome";
}
"Noronha";
{
  "exemplarCity";
  "Noronha";
}
"North_Dakota";
{
  "Beulah";
  {
    "exemplarCity";
    "Beulah, North Dakota";
  }
  "New_Salem";
  {
    "exemplarCity";
    "New Salem, North Dakota";
  }
}
```

```

    }
    "Center";
    {
        "exemplarCity";
        "Center, North Dakota";
    }
}
"Ojinaga";
{
    "exemplarCity";
    "Ojinaga";
}
"Panama";
{
    "exemplarCity";
    "Panama";
}
"Pangnirtung";
{
    "exemplarCity";
    "Pangnirtung";
}
"Paramaribo";
{
    "exemplarCity";
    "Paramaribo";
}
"Phoenix";
{
    "exemplarCity";
    "Phoenix";
}
"Port-au-Prince";
{
    "exemplarCity";
    "Port-au-Prince";
}
"Port_of_Spain";
{
    "exemplarCity";
    "Port of Spain";
}
"Porto_Velho";
{
    "exemplarCity";
    "Porto Velho";
}
"Puerto_Rico";
{
    "exemplarCity";
    "Puerto Rico";
}
"Rainy_River";
{
    "exemplarCity";
    "Rainy River";
}
}

```

```
"Rankin_Inlet";
{
  "exemplarCity";
  "Rankin Inlet";
}
"Recife";
{
  "exemplarCity";
  "Recife";
}
"Regina";
{
  "exemplarCity";
  "Regina";
}
"Resolute";
{
  "exemplarCity";
  "Resolute";
}
"Rio_Branco";
{
  "exemplarCity";
  "Rio Branco";
}
"Santa_Isabel";
{
  "exemplarCity";
  "Santa Isabel";
}
"Santarem";
{
  "exemplarCity";
  "Santarem";
}
"Santiago";
{
  "exemplarCity";
  "Santiago";
}
"Santo_Domingo";
{
  "exemplarCity";
  "Santo Domingo";
}
"Sao_Paulo";
{
  "exemplarCity";
  "São Paulo";
}
"Scoresbysund";
{
  "exemplarCity";
  "Ittoqqortoormiit";
}
"Sitka";
{
```

```
        "exemplarCity";
        "Sitka";
    }
    "St_Barthelemy";
    {
        "exemplarCity";
        "Saint-Barthélemy";
    }
    "St_Johns";
    {
        "exemplarCity";
        "St. John's";
    }
    "St_Kitts";
    {
        "exemplarCity";
        "St. Kitts";
    }
    "St_Lucia";
    {
        "exemplarCity";
        "St. Lucia";
    }
    "St_Thomas";
    {
        "exemplarCity";
        "St. Thomas";
    }
    "St_Vincent";
    {
        "exemplarCity";
        "St. Vincent";
    }
    "Swift_Current";
    {
        "exemplarCity";
        "Swift Current";
    }
    "Tegucigalpa";
    {
        "exemplarCity";
        "Tegucigalpa";
    }
    "Thule";
    {
        "exemplarCity";
        "Thule";
    }
    "Thunder_Bay";
    {
        "exemplarCity";
        "Thunder Bay";
    }
    "Tijuana";
    {
        "exemplarCity";
        "Tijuana";
    }
```



```
}
  "Toronto";
  {
    "exemplarCity";
    "Toronto";
  }
  "Tortola";
  {
    "exemplarCity";
    "Tortola";
  }
  "Vancouver";
  {
    "exemplarCity";
    "Vancouver";
  }
  "Whitehorse";
  {
    "exemplarCity";
    "Whitehorse";
  }
  "Winnipeg";
  {
    "exemplarCity";
    "Winnipeg";
  }
  "Yakutat";
  {
    "exemplarCity";
    "Yakutat";
  }
  "Yellowknife";
  {
    "exemplarCity";
    "Yellowknife";
  }
}
"Atlantic";
{
  "Azores";
  {
    "exemplarCity";
    "Azoren";
  }
  "Bermuda";
  {
    "exemplarCity";
    "Bermudas";
  }
  "Canary";
  {
    "exemplarCity";
    "Kanaren";
  }
  "Cape_Verde";
  {
    "exemplarCity";
```

```

        "Cabo Verde";
    }
    "Faeroe";
    {
        "exemplarCity";
        "Färöer";
    }
    "Madeira";
    {
        "exemplarCity";
        "Madeira";
    }
    "Reykjavik";
    {
        "exemplarCity";
        "Reykjavík";
    }
    "South_Georgia";
    {
        "exemplarCity";
        "Südgeorgien";
    }
    "St_Helena";
    {
        "exemplarCity";
        "St. Helena";
    }
    "Stanley";
    {
        "exemplarCity";
        "Stanley";
    }
}
"Europe";
{
    "Amsterdam";
    {
        "exemplarCity";
        "Amsterdam";
    }
    "Andorra";
    {
        "exemplarCity";
        "Andorra";
    }
    "Astrakhan";
    {
        "exemplarCity";
        "Astrachan";
    }
    "Athens";
    {
        "exemplarCity";
        "Athen";
    }
    "Belgrade";
    {

```

```

        "exemplarCity";
        "Belgrad";
    }
    "Berlin";
    {
        "exemplarCity";
        "Berlin";
    }
    "Bratislava";
    {
        "exemplarCity";
        "Bratislava";
    }
    "Brussels";
    {
        "exemplarCity";
        "Brüssel";
    }
    "Bucharest";
    {
        "exemplarCity";
        "Bukarest";
    }
    "Budapest";
    {
        "exemplarCity";
        "Budapest";
    }
    "Busingen";
    {
        "exemplarCity";
        "Büsingen";
    }
    "Chisinau";
    {
        "exemplarCity";
        "Kischinau";
    }
    "Copenhagen";
    {
        "exemplarCity";
        "Kopenhagen";
    }
    "Dublin";
    {
        "long";
        {
            "daylight";
            "Irische Sommerzeit";
        }
        "exemplarCity";
        "Dublin";
    }
    "Gibraltar";
    {
        "exemplarCity";
        "Gibraltar";
    }

```

```
}
"Guernsey";
{
  "exemplarCity";
  "Guernsey";
}
"Helsinki";
{
  "exemplarCity";
  "Helsinki";
}
"Isle_of_Man";
{
  "exemplarCity";
  "Isle of Man";
}
"Istanbul";
{
  "exemplarCity";
  "Istanbul";
}
"Jersey";
{
  "exemplarCity";
  "Jersey";
}
"Kaliningrad";
{
  "exemplarCity";
  "Kaliningrad";
}
"Kiev";
{
  "exemplarCity";
  "Kiew";
}
"Kirov";
{
  "exemplarCity";
  "Kirow";
}
"Lisbon";
{
  "exemplarCity";
  "Lissabon";
}
"Ljubljana";
{
  "exemplarCity";
  "Ljubljana";
}
"London";
{
  "long";
  {
    "daylight";
    "Britische Sommerzeit";
  }
}
```

```
    }  
    "exemplarCity";  
    "London";  
  }  
  "Luxembourg";  
  {  
    "exemplarCity";  
    "Luxemburg";  
  }  
  "Madrid";  
  {  
    "exemplarCity";  
    "Madrid";  
  }  
  "Malta";  
  {  
    "exemplarCity";  
    "Malta";  
  }  
  "Mariehamn";  
  {  
    "exemplarCity";  
    "Mariehamn";  
  }  
  "Minsk";  
  {  
    "exemplarCity";  
    "Minsk";  
  }  
  "Monaco";  
  {  
    "exemplarCity";  
    "Monaco";  
  }  
  "Moscow";  
  {  
    "exemplarCity";  
    "Moskau";  
  }  
  "Oslo";  
  {  
    "exemplarCity";  
    "Oslo";  
  }  
  "Paris";  
  {  
    "exemplarCity";  
    "Paris";  
  }  
  "Podgorica";  
  {  
    "exemplarCity";  
    "Podgorica";  
  }  
  "Prague";  
  {  
    "exemplarCity";
```

```
        "Prag";
    }
    "Riga";
    {
        "exemplarCity";
        "Riga";
    }
    "Rome";
    {
        "exemplarCity";
        "Rom";
    }
    "Samara";
    {
        "exemplarCity";
        "Samara";
    }
    "San_Marino";
    {
        "exemplarCity";
        "San Marino";
    }
    "Sarajevo";
    {
        "exemplarCity";
        "Sarajevo";
    }
    "Simferopol";
    {
        "exemplarCity";
        "Simferopol";
    }
    "Skopje";
    {
        "exemplarCity";
        "Skopje";
    }
    "Sofia";
    {
        "exemplarCity";
        "Sofia";
    }
    "Stockholm";
    {
        "exemplarCity";
        "Stockholm";
    }
    "Tallinn";
    {
        "exemplarCity";
        "Tallinn";
    }
    "Tirane";
    {
        "exemplarCity";
        "Tirana";
    }
}
```

```
"Ulyanovsk";
{
  "exemplarCity";
  "Uljanowsk";
}
"Uzhgorod";
{
  "exemplarCity";
  "Uschgorod";
}
"Vaduz";
{
  "exemplarCity";
  "Vaduz";
}
"Vatican";
{
  "exemplarCity";
  "Vatikan";
}
"Vienna";
{
  "exemplarCity";
  "Wien";
}
"Vilnius";
{
  "exemplarCity";
  "Vilnius";
}
"Volgograd";
{
  "exemplarCity";
  "Wolgograd";
}
"Warsaw";
{
  "exemplarCity";
  "Warschau";
}
"Zagreb";
{
  "exemplarCity";
  "Zagreb";
}
"Zaporozhye";
{
  "exemplarCity";
  "Saporischja";
}
"Zurich";
{
  "exemplarCity";
  "Zürich";
}
}
"Africa";
```

```
{
  "Abidjan";
  {
    "exemplarCity";
    "Abidjan";
  }
  "Accra";
  {
    "exemplarCity";
    "Accra";
  }
  "Addis_Ababa";
  {
    "exemplarCity";
    "Addis Abeba";
  }
  "Algiers";
  {
    "exemplarCity";
    "Algier";
  }
  "Asmera";
  {
    "exemplarCity";
    "Asmara";
  }
  "Bamako";
  {
    "exemplarCity";
    "Bamako";
  }
  "Bangui";
  {
    "exemplarCity";
    "Bangui";
  }
  "Banjul";
  {
    "exemplarCity";
    "Banjul";
  }
  "Bissau";
  {
    "exemplarCity";
    "Bissau";
  }
  "Blantyre";
  {
    "exemplarCity";
    "Blantyre";
  }
  "Brazzaville";
  {
    "exemplarCity";
    "Brazzaville";
  }
  "Bujumbura";
```



```
{
    "exemplarCity";
    "Bujumbura";
}
"Cairo";
{
    "exemplarCity";
    "Kairo";
}
"Casablanca";
{
    "exemplarCity";
    "Casablanca";
}
"Ceuta";
{
    "exemplarCity";
    "Ceuta";
}
"Conakry";
{
    "exemplarCity";
    "Conakry";
}
"Dakar";
{
    "exemplarCity";
    "Dakar";
}
"Dar_es_Salaam";
{
    "exemplarCity";
    "Daressalam";
}
"Djibouti";
{
    "exemplarCity";
    "Dschibuti";
}
"Douala";
{
    "exemplarCity";
    "Douala";
}
"El_Aaiun";
{
    "exemplarCity";
    "El Aaiún";
}
"Freetown";
{
    "exemplarCity";
    "Freetown";
}
"Gaborone";
{
    "exemplarCity";
```

```
        "Gaborone";
    }
    "Harare";
    {
        "exemplarCity";
        "Harare";
    }
    "Johannesburg";
    {
        "exemplarCity";
        "Johannesburg";
    }
    "Juba";
    {
        "exemplarCity";
        "Juba";
    }
    "Kampala";
    {
        "exemplarCity";
        "Kampala";
    }
    "Khartoum";
    {
        "exemplarCity";
        "Khartum";
    }
    "Kigali";
    {
        "exemplarCity";
        "Kigali";
    }
    "Kinshasa";
    {
        "exemplarCity";
        "Kinshasa";
    }
    "Lagos";
    {
        "exemplarCity";
        "Lagos";
    }
    "Libreville";
    {
        "exemplarCity";
        "Libreville";
    }
    "Lome";
    {
        "exemplarCity";
        "Lomé";
    }
    "Luanda";
    {
        "exemplarCity";
        "Luanda";
    }
}
```

```
"Lubumbashi";
{
  "exemplarCity";
  "Lubumbashi";
}
"Lusaka";
{
  "exemplarCity";
  "Lusaka";
}
"Malabo";
{
  "exemplarCity";
  "Malabo";
}
"Maputo";
{
  "exemplarCity";
  "Maputo";
}
"Maseru";
{
  "exemplarCity";
  "Maseru";
}
"Mbabane";
{
  "exemplarCity";
  "Mbabane";
}
"Mogadishu";
{
  "exemplarCity";
  "Mogadishu";
}
"Monrovia";
{
  "exemplarCity";
  "Monrovia";
}
"Nairobi";
{
  "exemplarCity";
  "Nairobi";
}
"Ndjamena";
{
  "exemplarCity";
  "N' Djamena";
}
"Niamey";
{
  "exemplarCity";
  "Niamey";
}
"Nouakchott";
{
```

```

        "exemplarCity";
        "Nouakchott";
    }
    "Ouagadougou";
    {
        "exemplarCity";
        "Ouagadougou";
    }
    "Porto-Novo";
    {
        "exemplarCity";
        "Porto Novo";
    }
    "Sao_Tome";
    {
        "exemplarCity";
        "São Tomé";
    }
    "Tripoli";
    {
        "exemplarCity";
        "Tripolis";
    }
    "Tunis";
    {
        "exemplarCity";
        "Tunis";
    }
    "Windhoek";
    {
        "exemplarCity";
        "Windhoek";
    }
}
"Asia";
{
    "Aden";
    {
        "exemplarCity";
        "Aden";
    }
    "Almaty";
    {
        "exemplarCity";
        "Almaty";
    }
    "Amman";
    {
        "exemplarCity";
        "Amman";
    }
    "Anadyr";
    {
        "exemplarCity";
        "Anadyr";
    }
    "Aqtau";

```

```
{
    "exemplarCity";
    "Aqtau";
}
"Aqtobe";
{
    "exemplarCity";
    "Aktobe";
}
"Ashgabat";
{
    "exemplarCity";
    "Aşgabat";
}
"Baghdad";
{
    "exemplarCity";
    "Bagdad";
}
"Bahrain";
{
    "exemplarCity";
    "Bahrain";
}
"Baku";
{
    "exemplarCity";
    "Baku";
}
"Bangkok";
{
    "exemplarCity";
    "Bangkok";
}
"Barnaul";
{
    "exemplarCity";
    "Barnaul";
}
"Beirut";
{
    "exemplarCity";
    "Beirut";
}
"Bishkek";
{
    "exemplarCity";
    "Bischkek";
}
"Brunei";
{
    "exemplarCity";
    "Brunei";
}
"Calcutta";
{
    "exemplarCity";
```

```
        "Kalkutta";
    }
    "Chita";
    {
        "exemplarCity";
        "Tschita";
    }
    "Choibalsan";
    {
        "exemplarCity";
        "Tschoibalsan";
    }
    "Colombo";
    {
        "exemplarCity";
        "Colombo";
    }
    "Damascus";
    {
        "exemplarCity";
        "Damaskus";
    }
    "Dhaka";
    {
        "exemplarCity";
        "Dhaka";
    }
    "Dili";
    {
        "exemplarCity";
        "Dili";
    }
    "Dubai";
    {
        "exemplarCity";
        "Dubai";
    }
    "Dushanbe";
    {
        "exemplarCity";
        "Duschanbe";
    }
    "Gaza";
    {
        "exemplarCity";
        "Gaza";
    }
    "Hebron";
    {
        "exemplarCity";
        "Hebron";
    }
    "Hong_Kong";
    {
        "exemplarCity";
        "Hongkong";
    }
}
```

```
"Hovd";
{
  "exemplarCity";
  "Chowd";
}
"Irkutsk";
{
  "exemplarCity";
  "Irkutsk";
}
"Jakarta";
{
  "exemplarCity";
  "Jakarta";
}
"Jayapura";
{
  "exemplarCity";
  "Jayapura";
}
"Jerusalem";
{
  "exemplarCity";
  "Jerusalem";
}
"Kabul";
{
  "exemplarCity";
  "Kabul";
}
"Kamchatka";
{
  "exemplarCity";
  "Kamtschatka";
}
"Karachi";
{
  "exemplarCity";
  "Karatschi";
}
"Katmandu";
{
  "exemplarCity";
  "Kathmandu";
}
"Khandyga";
{
  "exemplarCity";
  "Chandyga";
}
"Krasnoyarsk";
{
  "exemplarCity";
  "Krasnojarsk";
}
"Kuala_Lumpur";
{
```

```
        "exemplarCity";
        "Kuala Lumpur";
    }
    "Kuching";
    {
        "exemplarCity";
        "Kuching";
    }
    "Kuwait";
    {
        "exemplarCity";
        "Kuwait";
    }
    "Macau";
    {
        "exemplarCity";
        "Macao";
    }
    "Magadan";
    {
        "exemplarCity";
        "Magadan";
    }
    "Makassar";
    {
        "exemplarCity";
        "Makassar";
    }
    "Manila";
    {
        "exemplarCity";
        "Manila";
    }
    "Muscat";
    {
        "exemplarCity";
        "Maskat";
    }
    "Nicosia";
    {
        "exemplarCity";
        "Nikosia";
    }
    "Novokuznetsk";
    {
        "exemplarCity";
        "Nowokuznetsk";
    }
    "Novosibirsk";
    {
        "exemplarCity";
        "Nowosibirsk";
    }
    "Omsk";
    {
        "exemplarCity";
        "Omsk";
    }
}
```



```
}
"Oral";
{
  "exemplarCity";
  "Oral";
}
"Phnom_Penh";
{
  "exemplarCity";
  "Phnom Penh";
}
"Pontianak";
{
  "exemplarCity";
  "Pontianak";
}
"Pyongyang";
{
  "exemplarCity";
  "Pjöngjang";
}
"Qatar";
{
  "exemplarCity";
  "Katar";
}
"Qyzylorda";
{
  "exemplarCity";
  "Qysylorda";
}
"Rangoon";
{
  "exemplarCity";
  "Rangun";
}
"Riyadh";
{
  "exemplarCity";
  "Riad";
}
"Saigon";
{
  "exemplarCity";
  "Ho-Chi-Minh-Stadt";
}
"Sakhalin";
{
  "exemplarCity";
  "Sachalin";
}
"Samarkand";
{
  "exemplarCity";
  "Samarkand";
}
"Seoul";
```

```
{
    "exemplarCity";
    "Seoul";
}
"Shanghai";
{
    "exemplarCity";
    "Shanghai";
}
"Singapore";
{
    "exemplarCity";
    "Singapur";
}
"Srednekolymsk";
{
    "exemplarCity";
    "Srednekolymsk";
}
"Taipei";
{
    "exemplarCity";
    "Taipeh";
}
"Tashkent";
{
    "exemplarCity";
    "Taschkent";
}
"Tbilisi";
{
    "exemplarCity";
    "Tiflis";
}
"Tehran";
{
    "exemplarCity";
    "Teheran";
}
"Thimphu";
{
    "exemplarCity";
    "Thimphu";
}
"Tokyo";
{
    "exemplarCity";
    "Tokio";
}
"Tomsk";
{
    "exemplarCity";
    "Tomsk";
}
"Ulaanbaatar";
{
    "exemplarCity";
```

```
        "Ulaanbaatar";
    }
    "Urumqi";
    {
        "exemplarCity";
        "Ürümqi";
    }
    "Ust-Nera";
    {
        "exemplarCity";
        "Ust-Nera";
    }
    "Vientiane";
    {
        "exemplarCity";
        "Vientiane";
    }
    "Vladivostok";
    {
        "exemplarCity";
        "Wladiwostok";
    }
    "Yakutsk";
    {
        "exemplarCity";
        "Jakutsk";
    }
    "Yekaterinburg";
    {
        "exemplarCity";
        "Jekaterinburg";
    }
    "Yerevan";
    {
        "exemplarCity";
        "Eriwan";
    }
}
"Indian";
{
    "Antananarivo";
    {
        "exemplarCity";
        "Antananarivo";
    }
    "Chagos";
    {
        "exemplarCity";
        "Chagos";
    }
    "Christmas";
    {
        "exemplarCity";
        "Weihnachtsinsel";
    }
    "Cocos";
    {
```

```

        "exemplarCity";
        "Cocos";
    }
    "Comoro";
    {
        "exemplarCity";
        "Komoren";
    }
    "Kerguelen";
    {
        "exemplarCity";
        "Kerguelen";
    }
    "Mahe";
    {
        "exemplarCity";
        "Mahe";
    }
    "Maldives";
    {
        "exemplarCity";
        "Malediven";
    }
    "Mauritius";
    {
        "exemplarCity";
        "Mauritius";
    }
    "Mayotte";
    {
        "exemplarCity";
        "Mayotte";
    }
    "Reunion";
    {
        "exemplarCity";
        "Réunion";
    }
}
"Australia";
{
    "Adelaide";
    {
        "exemplarCity";
        "Adelaide";
    }
    "Brisbane";
    {
        "exemplarCity";
        "Brisbane";
    }
    "Broken_Hill";
    {
        "exemplarCity";
        "Broken Hill";
    }
    "Currie";

```

```
{
    "exemplarCity";
    "Currie";
}
"Darwin";
{
    "exemplarCity";
    "Darwin";
}
"Eucla";
{
    "exemplarCity";
    "Eucla";
}
"Hobart";
{
    "exemplarCity";
    "Hobart";
}
"Lindeman";
{
    "exemplarCity";
    "Lindeman";
}
"Lord_Howe";
{
    "exemplarCity";
    "Lord Howe";
}
"Melbourne";
{
    "exemplarCity";
    "Melbourne";
}
"Perth";
{
    "exemplarCity";
    "Perth";
}
"Sydney";
{
    "exemplarCity";
    "Sydney";
}
}
"Pacific";
{
    "Apia";
    {
        "exemplarCity";
        "Apia";
    }
    "Auckland";
    {
        "exemplarCity";
        "Auckland";
    }
}
```

```
"Bougainville";
{
  "exemplarCity";
  "Bougainville";
}
"Chatham";
{
  "exemplarCity";
  "Chatham";
}
"Easter";
{
  "exemplarCity";
  "Osterinsel";
}
"Efate";
{
  "exemplarCity";
  "Efate";
}
"Enderbury";
{
  "exemplarCity";
  "Enderbury";
}
"Fakaofo";
{
  "exemplarCity";
  "Fakaofo";
}
"Fiji";
{
  "exemplarCity";
  "Fidschi";
}
"Funafuti";
{
  "exemplarCity";
  "Funafuti";
}
"Galapagos";
{
  "exemplarCity";
  "Galapagos";
}
"Gambier";
{
  "exemplarCity";
  "Gambier";
}
"Guadalcanal";
{
  "exemplarCity";
  "Guadalcanal";
}
"Guam";
{
```

```
        "exemplarCity";
        "Guam";
    }
    "Honolulu";
    {
        "exemplarCity";
        "Honolulu";
    }
    "Johnston";
    {
        "exemplarCity";
        "Johnston";
    }
    "Kiritimati";
    {
        "exemplarCity";
        "Kiritimati";
    }
    "Kosrae";
    {
        "exemplarCity";
        "Kosrae";
    }
    "Kwajalein";
    {
        "exemplarCity";
        "Kwajalein";
    }
    "Majuro";
    {
        "exemplarCity";
        "Majuro";
    }
    "Marquesas";
    {
        "exemplarCity";
        "Marquesas";
    }
    "Midway";
    {
        "exemplarCity";
        "Midway";
    }
    "Nauru";
    {
        "exemplarCity";
        "Nauru";
    }
    "Niue";
    {
        "exemplarCity";
        "Niue";
    }
    "Norfolk";
    {
        "exemplarCity";
        "Norfolk";
    }
}
```

```
}
"Noumea";
{
  "exemplarCity";
  "Noumea";
}
"Pago_Pago";
{
  "exemplarCity";
  "Pago Pago";
}
"Palau";
{
  "exemplarCity";
  "Palau";
}
"Pitcairn";
{
  "exemplarCity";
  "Pitcairn";
}
"Ponape";
{
  "exemplarCity";
  "Pohnpei";
}
"Port_Moresby";
{
  "exemplarCity";
  "Port Moresby";
}
"Rarotonga";
{
  "exemplarCity";
  "Rarotonga";
}
"Saipan";
{
  "exemplarCity";
  "Saipan";
}
"Tahiti";
{
  "exemplarCity";
  "Tahiti";
}
"Tarawa";
{
  "exemplarCity";
  "Tarawa";
}
"Tongatapu";
{
  "exemplarCity";
  "Tongatapu";
}
"Truk";
```



```

        {
            "exemplarCity";
            "Chuuk";
        }
        "Wake";
        {
            "exemplarCity";
            "Wake";
        }
        "Wallis";
        {
            "exemplarCity";
            "Wallis";
        }
    }
    "Arctic";
    {
        "Longyearbyen";
        {
            "exemplarCity";
            "Longyearbyen";
        }
    }
    "Antarctica";
    {
        "Casey";
        {
            "exemplarCity";
            "Casey";
        }
        "Davis";
        {
            "exemplarCity";
            "Davis";
        }
        "DumontDUrville";
        {
            "exemplarCity";
            "Dumont d'Urville";
        }
        "Macquarie";
        {
            "exemplarCity";
            "Macquarie";
        }
        "Mawson";
        {
            "exemplarCity";
            "Mawson";
        }
        "McMurdo";
        {
            "exemplarCity";
            "McMurdo";
        }
        "Palmer";
        {

```

```
        "exemplarCity";
        "Palmer";
    }
    "Rothera";
    {
        "exemplarCity";
        "Rothera";
    }
    "Syowa";
    {
        "exemplarCity";
        "Syowa";
    }
    "Troll";
    {
        "exemplarCity";
        "Troll";
    }
    "Vostok";
    {
        "exemplarCity";
        "Wostok";
    }
}
"Etc";
{
    "GMT";
    {
        "exemplarCity";
        "GMT";
    }
    "GMT1";
    {
        "exemplarCity";
        "GMT+1";
    }
    "GMT10";
    {
        "exemplarCity";
        "GMT+10";
    }
    "GMT11";
    {
        "exemplarCity";
        "GMT+11";
    }
    "GMT12";
    {
        "exemplarCity";
        "GMT+12";
    }
    "GMT2";
    {
        "exemplarCity";
        "GMT+2";
    }
    "GMT3";
```

```
{
    "exemplarCity";
    "GMT+3";
}
"GMT4";
{
    "exemplarCity";
    "GMT+4";
}
"GMT5";
{
    "exemplarCity";
    "GMT+5";
}
"GMT6";
{
    "exemplarCity";
    "GMT+6";
}
"GMT7";
{
    "exemplarCity";
    "GMT+7";
}
"GMT8";
{
    "exemplarCity";
    "GMT+8";
}
"GMT9";
{
    "exemplarCity";
    "GMT+9";
}
"GMT-1";
{
    "exemplarCity";
    "GMT-1";
}
"GMT-10";
{
    "exemplarCity";
    "GMT-10";
}
"GMT-11";
{
    "exemplarCity";
    "GMT-11";
}
"GMT-12";
{
    "exemplarCity";
    "GMT-12";
}
"GMT-13";
{
    "exemplarCity";
```

```

        "GMT-13";
    }
    "GMT-14";
    {
        "exemplarCity";
        "GMT-14";
    }
    "GMT-2";
    {
        "exemplarCity";
        "GMT-2";
    }
    "GMT-3";
    {
        "exemplarCity";
        "GMT-3";
    }
    "GMT-4";
    {
        "exemplarCity";
        "GMT-4";
    }
    "GMT-5";
    {
        "exemplarCity";
        "GMT-5";
    }
    "GMT-6";
    {
        "exemplarCity";
        "GMT-6";
    }
    "GMT-7";
    {
        "exemplarCity";
        "GMT-7";
    }
    "GMT-8";
    {
        "exemplarCity";
        "GMT-8";
    }
    "GMT-9";
    {
        "exemplarCity";
        "GMT-9";
    }
    "Unknown";
    {
        "exemplarCity";
        "Unbekannt";
    }
    }
}
"metazone";
{
    "Acre";

```

```

    {
      "long";
      {
        "generic";
        "Acre-Zeit",
        "standard";
        "Acre-Normalzeit",
        "daylight";
        "Acre-Sommerzeit";
      }
    }
    "Afghanistan";
    {
      "long";
      {
        "standard";
        "Afghanistan-Zeit";
      }
    }
    "Africa_Central";
    {
      "long";
      {
        "standard";
        "Zentralafrikanische Zeit";
      }
    }
    "Africa_Eastern";
    {
      "long";
      {
        "standard";
        "Ostafrikanische Zeit";
      }
    }
    "Africa_Southern";
    {
      "long";
      {
        "standard";
        "Südafrikanische Zeit";
      }
    }
    "Africa_Western";
    {
      "long";
      {
        "generic";
        "Westafrikanische Zeit",
        "standard";
        "Westafrikanische Normalzeit",
        "daylight";
        "Westafrikanische Sommerzeit";
      }
    }
    "Alaska";
    {

```

```

        "long";
        {
            "generic";
            "Alaska-Zeit",
            "standard";
            "Alaska-Normalzeit",
            "daylight";
            "Alaska-Sommerzeit";
        }
    }
    "Almaty";
    {
        "long";
        {
            "generic";
            "Almaty-Zeit",
            "standard";
            "Almaty-Normalzeit",
            "daylight";
            "Almaty-Sommerzeit";
        }
    }
    "Amazon";
    {
        "long";
        {
            "generic";
            "Amazonas-Zeit",
            "standard";
            "Amazonas-Normalzeit",
            "daylight";
            "Amazonas-Sommerzeit";
        }
    }
    "America_Central";
    {
        "long";
        {
            "generic";
            "Nordamerikanische Inlandzeit",
            "standard";
            "Nordamerikanische Inland-Normalzeit",
            "daylight";
            "Nordamerikanische Inland-Sommerzeit";
        }
    }
    "America_Eastern";
    {
        "long";
        {
            "generic";
            "Nordamerikanische Ostküstenzeit",
            "standard";
            "Nordamerikanische Ostküsten-Normalzeit",
            "daylight";
            "Nordamerikanische Ostküsten-Sommerzeit";
        }
    }

```

```

    }
    "America_Mountain";
    {
        "long";
        {
            "generic";
            "Rocky-Mountain-Zeit",
            "standard";
            "Rocky Mountain-Normalzeit",
            "daylight";
            "Rocky-Mountain-Sommerzeit";
        }
    }
    "America_Pacific";
    {
        "long";
        {
            "generic";
            "Nordamerikanische Westküstenzeit",
            "standard";
            "Nordamerikanische Westküsten-Normalzeit",
            "daylight";
            "Nordamerikanische Westküsten-Sommerzeit";
        }
    }
    "Anadyr";
    {
        "long";
        {
            "generic";
            "Anadyr Zeit",
            "standard";
            "Anadyr Normalzeit",
            "daylight";
            "Anadyr Sommerzeit";
        }
    }
    "Apia";
    {
        "long";
        {
            "generic";
            "Apia-Zeit",
            "standard";
            "Apia-Normalzeit",
            "daylight";
            "Apia-Sommerzeit";
        }
    }
    "Aqtau";
    {
        "long";
        {
            "generic";
            "Aqtau-Zeit",
            "standard";
            "Aqtau-Normalzeit",

```

```

        "daylight";
        "Aqtau-Sommerzeit";
    }
}
"Aqtobe";
{
    "long";
    {
        "generic";
        "Aqtöbe-Zeit",
        "standard";
        "Aqtöbe-Normalzeit",
        "daylight";
        "Aqtöbe-Sommerzeit";
    }
}
"Arabian";
{
    "long";
    {
        "generic";
        "Arabische Zeit",
        "standard";
        "Arabische Normalzeit",
        "daylight";
        "Arabische Sommerzeit";
    }
}
"Argentina";
{
    "long";
    {
        "generic";
        "Argentinische Zeit",
        "standard";
        "Argentinische Normalzeit",
        "daylight";
        "Argentinische Sommerzeit";
    }
}
"Argentina_Western";
{
    "long";
    {
        "generic";
        "Westargentinische Zeit",
        "standard";
        "Westargentinische Normalzeit",
        "daylight";
        "Westargentinische Sommerzeit";
    }
}
"Armenia";
{
    "long";
    {
        "generic";

```



```

        "Armenische Zeit",
        "standard";
        "Armenische Normalzeit",
        "daylight";
        "Armenische Sommerzeit";
    }
}
"Atlantic";
{
    "long";
    {
        "generic";
        "Atlantik-Zeit",
        "standard";
        "Atlantik-Normalzeit",
        "daylight";
        "Atlantik-Sommerzeit";
    }
}
"Australia_Central";
{
    "long";
    {
        "generic";
        "Zentralaustralische Zeit",
        "standard";
        "Zentralaustralische Normalzeit",
        "daylight";
        "Zentralaustralische Sommerzeit";
    }
}
"Australia_CentralWestern";
{
    "long";
    {
        "generic";
        "Zentral-/Westaustralische Zeit",
        "standard";
        "Zentral-/Westaustralische Normalzeit",
        "daylight";
        "Zentral-/Westaustralische Sommerzeit";
    }
}
"Australia_Eastern";
{
    "long";
    {
        "generic";
        "Ostaustralische Zeit",
        "standard";
        "Ostaustralische Normalzeit",
        "daylight";
        "Ostaustralische Sommerzeit";
    }
}
"Australia_Western";
{

```

```

        "long";
        {
            "generic";
            "Westaustralische Zeit",
            "standard";
            "Westaustralische Normalzeit",
            "daylight";
            "Westaustralische Sommerzeit";
        }
    }
    "Azerbaijan";
    {
        "long";
        {
            "generic";
            "Aserbajdschanische Zeit",
            "standard";
            "Aserbajdschanische Normalzeit",
            "daylight";
            "Aserbajdschanische Sommerzeit";
        }
    }
    "Azores";
    {
        "long";
        {
            "generic";
            "Azoren-Zeit",
            "standard";
            "Azoren-Normalzeit",
            "daylight";
            "Azoren-Sommerzeit";
        }
    }
    "Bangladesh";
    {
        "long";
        {
            "generic";
            "Bangladesch-Zeit",
            "standard";
            "Bangladesch-Normalzeit",
            "daylight";
            "Bangladesch-Sommerzeit";
        }
    }
    "Bhutan";
    {
        "long";
        {
            "standard";
            "Bhutan-Zeit";
        }
    }
    "Bolivia";
    {
        "long";

```

```

        {
            "standard";
            "Bolivianische Zeit";
        }
    }
    "Brasilia";
    {
        "long";
        {
            "generic";
            "Brasília-Zeit",
            "standard";
            "Brasília-Normalzeit",
            "daylight";
            "Brasília-Sommerzeit";
        }
    }
    "Brunei";
    {
        "long";
        {
            "standard";
            "Brunei-Zeit";
        }
    }
    "Cape_Verde";
    {
        "long";
        {
            "generic";
            "Cabo-Verde-Zeit",
            "standard";
            "Cabo-Verde-Normalzeit",
            "daylight";
            "Cabo-Verde-Sommerzeit";
        }
    }
    "Casey";
    {
        "long";
        {
            "standard";
            "Casey-Zeit";
        }
    }
    "Chamorro";
    {
        "long";
        {
            "standard";
            "Chamorro-Zeit";
        }
    }
    "Chatham";
    {
        "long";
        {

```

```

        "generic";
        "Chatham-Zeit",
        "standard";
        "Chatham-Normalzeit",
        "daylight";
        "Chatham-Sommerzeit";
    }
}
"Chile";
{
    "long";
    {
        "generic";
        "Chilenische Zeit",
        "standard";
        "Chilenische Normalzeit",
        "daylight";
        "Chilenische Sommerzeit";
    }
}
"China";
{
    "long";
    {
        "generic";
        "Chinesische Zeit",
        "standard";
        "Chinesische Normalzeit",
        "daylight";
        "Chinesische Sommerzeit";
    }
}
"Choibalsan";
{
    "long";
    {
        "generic";
        "Tschoibalsan-Zeit",
        "standard";
        "Tschoibalsan-Normalzeit",
        "daylight";
        "Tschoibalsan-Sommerzeit";
    }
}
"Christmas";
{
    "long";
    {
        "standard";
        "Weihnachtsinsel-Zeit";
    }
}
"Cocos";
{
    "long";
    {
        "standard";
    }
}

```

```

        "Kokosinseln-Zeit";
    }
}
"Colombia";
{
    "long";
    {
        "generic";
        "Kolumbianische Zeit",
        "standard";
        "Kolumbianische Normalzeit",
        "daylight";
        "Kolumbianische Sommerzeit";
    }
}
"Cook";
{
    "long";
    {
        "generic";
        "Cookinseln-Zeit",
        "standard";
        "Cookinseln-Normalzeit",
        "daylight";
        "Cookinseln-Sommerzeit";
    }
}
"Cuba";
{
    "long";
    {
        "generic";
        "Kubanische Zeit",
        "standard";
        "Kubanische Normalzeit",
        "daylight";
        "Kubanische Sommerzeit";
    }
}
"Davis";
{
    "long";
    {
        "standard";
        "Davis-Zeit";
    }
}
"DumontDUrville";
{
    "long";
    {
        "standard";
        "Dumont-d'Urville-Zeit";
    }
}
"East_Timor";
{

```

```

        "long";
        {
            "standard";
            "Osttimor-Zeit";
        }
    }
    "Easter";
    {
        "long";
        {
            "generic";
            "Osterinsel-Zeit",
            "standard";
            "Osterinsel-Normalzeit",
            "daylight";
            "Osterinsel-Sommerzeit";
        }
    }
    "Ecuador";
    {
        "long";
        {
            "standard";
            "Ecuadorianische Zeit";
        }
    }
    "Europe_Central";
    {
        "long";
        {
            "generic";
            "Mittleuropäische Zeit",
            "standard";
            "Mittleuropäische Normalzeit",
            "daylight";
            "Mittleuropäische Sommerzeit";
        }
        "short";
        {
            "generic";
            "MEZ",
            "standard";
            "MEZ",
            "daylight";
            "MESZ";
        }
    }
    "Europe_Eastern";
    {
        "long";
        {
            "generic";
            "Osteuropäische Zeit",
            "standard";
            "Osteuropäische Normalzeit",
            "daylight";
            "Osteuropäische Sommerzeit";
        }
    }

```

```

    }
    "short";
    {
        "generic";
        "OEZ",
        "standard";
        "OEZ",
        "daylight";
        "OESZ";
    }
}
"Europe_Further_Eastern";
{
    "long";
    {
        "standard";
        "Kaliningrader Zeit";
    }
}
"Europe_Western";
{
    "long";
    {
        "generic";
        "Westeuropäische Zeit",
        "standard";
        "Westeuropäische Normalzeit",
        "daylight";
        "Westeuropäische Sommerzeit";
    }
    "short";
    {
        "generic";
        "WEZ",
        "standard";
        "WEZ",
        "daylight";
        "WESZ";
    }
}
"Falkland";
{
    "long";
    {
        "generic";
        "Falklandinseln-Zeit",
        "standard";
        "Falklandinseln-Normalzeit",
        "daylight";
        "Falklandinseln-Sommerzeit";
    }
}
"Fiji";
{
    "long";
    {
        "generic";

```

```

        "Fidschi-Zeit",
        "standard";
        "Fidschi-Normalzeit",
        "daylight";
        "Fidschi-Sommerzeit";
    }
}
"French_Guiana";
{
    "long";
    {
        "standard";
        "Französisch-Guayana-Zeit";
    }
}
"French_Southern";
{
    "long";
    {
        "standard";
        "Französische Süd- und Antarktisgebiete-
Zeit";
    }
}
"Galapagos";
{
    "long";
    {
        "standard";
        "Galapagos-Zeit";
    }
}
"Gambier";
{
    "long";
    {
        "standard";
        "Gambier-Zeit";
    }
}
"Georgia";
{
    "long";
    {
        "generic";
        "Georgische Zeit",
        "standard";
        "Georgische Normalzeit",
        "daylight";
        "Georgische Sommerzeit";
    }
}
"Gilbert_Islands";
{
    "long";
    {
        "standard";

```



```

        "Gilbert-Inseln-Zeit";
    }
}
"GMT";
{
    "long";
    {
        "standard";
        "Mittlere Greenwich-Zeit";
    }
}
"Greenland_Eastern";
{
    "long";
    {
        "generic";
        "Ostgrönland-Zeit",
        "standard";
        "Ostgrönland-Normalzeit",
        "daylight";
        "Ostgrönland-Sommerzeit";
    }
}
"Greenland_Western";
{
    "long";
    {
        "generic";
        "Westgrönland-Zeit",
        "standard";
        "Westgrönland-Normalzeit",
        "daylight";
        "Westgrönland-Sommerzeit";
    }
}
"Guam";
{
    "long";
    {
        "standard";
        "Guam-Zeit";
    }
}
"Gulf";
{
    "long";
    {
        "standard";
        "Golf-Zeit";
    }
}
"Guyana";
{
    "long";
    {
        "standard";
        "Guyana-Zeit";
    }
}

```

```

    }
  }
  "Hawaii_Aleutian";
  {
    "long";
    {
      "generic";
      "Hawaii-Aleuten-Zeit",
      "standard";
      "Hawaii-Aleuten-Normalzeit",
      "daylight";
      "Hawaii-Aleuten-Sommerzeit";
    }
  }
  "Hong_Kong";
  {
    "long";
    {
      "generic";
      "Hongkong-Zeit",
      "standard";
      "Hongkong-Normalzeit",
      "daylight";
      "Hongkong-Sommerzeit";
    }
  }
  "Hovd";
  {
    "long";
    {
      "generic";
      "Chowd-Zeit",
      "standard";
      "Chowd-Normalzeit",
      "daylight";
      "Chowd-Sommerzeit";
    }
  }
  "India";
  {
    "long";
    {
      "standard";
      "Indische Zeit";
    }
  }
  "Indian_Ocean";
  {
    "long";
    {
      "standard";
      "Indischer Ozean-Zeit";
    }
  }
  "Indochina";
  {
    "long";

```

```

        {
            "standard";
            "Indochina-Zeit";
        }
    }
    "Indonesia_Central";
    {
        "long";
        {
            "standard";
            "Zentralindonesische Zeit";
        }
    }
    "Indonesia_Eastern";
    {
        "long";
        {
            "standard";
            "Ostindonesische Zeit";
        }
    }
    "Indonesia_Western";
    {
        "long";
        {
            "standard";
            "Westindonesische Zeit";
        }
    }
    "Iran";
    {
        "long";
        {
            "generic";
            "Iranische Zeit",
            "standard";
            "Iranische Normalzeit",
            "daylight";
            "Iranische Sommerzeit";
        }
    }
    "Irkutsk";
    {
        "long";
        {
            "generic";
            "Irkutsk-Zeit",
            "standard";
            "Irkutsk-Normalzeit",
            "daylight";
            "Irkutsk-Sommerzeit";
        }
    }
    "Israel";
    {
        "long";
        {

```

```

        "generic";
        "Israelische Zeit",
        "standard";
        "Israelische Normalzeit",
        "daylight";
        "Israelische Sommerzeit";
    }
}
"Japan";
{
    "long";
    {
        "generic";
        "Japanische Zeit",
        "standard";
        "Japanische Normalzeit",
        "daylight";
        "Japanische Sommerzeit";
    }
}
"Kamchatka";
{
    "long";
    {
        "generic";
        "Kamtschatka-Zeit",
        "standard";
        "Kamtschatka-Normalzeit",
        "daylight";
        "Kamtschatka-Sommerzeit";
    }
}
"Kazakhstan_Eastern";
{
    "long";
    {
        "standard";
        "Ostkasachische Zeit";
    }
}
"Kazakhstan_Western";
{
    "long";
    {
        "standard";
        "Westkasachische Zeit";
    }
}
"Korea";
{
    "long";
    {
        "generic";
        "Koreanische Zeit",
        "standard";
        "Koreanische Normalzeit",
        "daylight";
    }
}

```

```

        "Koreanische Sommerzeit";
    }
}
"Kosrae";
{
    "long";
    {
        "standard";
        "Kosrae-Zeit";
    }
}
"Krasnoyarsk";
{
    "long";
    {
        "generic";
        "Krasnojarsk-Zeit",
        "standard";
        "Krasnojarsk-Normalzeit",
        "daylight";
        "Krasnojarsk-Sommerzeit";
    }
}
"Kyrgystan";
{
    "long";
    {
        "standard";
        "Kirgisistan-Zeit";
    }
}
"Lanka";
{
    "long";
    {
        "standard";
        "Sri-Lanka-Zeit";
    }
}
"Line_Islands";
{
    "long";
    {
        "standard";
        "Linieninseln-Zeit";
    }
}
"Lord_Howe";
{
    "long";
    {
        "generic";
        "Lord-Howe-Zeit",
        "standard";
        "Lord-Howe-Normalzeit",
        "daylight";
        "Lord-Howe-Sommerzeit";
    }
}

```

```

    }
  }
  "Macau";
  {
    "long";
    {
      "generic";
      "Macau-Zeit",
      "standard";
      "Macau-Normalzeit",
      "daylight";
      "Macau-Sommerzeit";
    }
  }
  "Macquarie";
  {
    "long";
    {
      "standard";
      "Macquarieinsel-Zeit";
    }
  }
  "Magadan";
  {
    "long";
    {
      "generic";
      "Magadan-Zeit",
      "standard";
      "Magadan-Normalzeit",
      "daylight";
      "Magadan-Sommerzeit";
    }
  }
  "Malaysia";
  {
    "long";
    {
      "standard";
      "Malaysische Zeit";
    }
  }
  "Maldives";
  {
    "long";
    {
      "standard";
      "Malediven-Zeit";
    }
  }
  "Marquesas";
  {
    "long";
    {
      "standard";
      "Marquesas-Zeit";
    }
  }

```

```

    }
    "Marshall_Islands";
    {
        "long";
        {
            "standard";
            "Marshallinseln-Zeit";
        }
    }
    "Mauritius";
    {
        "long";
        {
            "generic";
            "Mauritius-Zeit",
            "standard";
            "Mauritius-Normalzeit",
            "daylight";
            "Mauritius-Sommerzeit";
        }
    }
    "Mawson";
    {
        "long";
        {
            "standard";
            "Mawson-Zeit";
        }
    }
    "Mexico_Northwest";
    {
        "long";
        {
            "generic";
            "Mexiko Nordwestliche Zone-Zeit",
            "standard";
            "Mexiko Nordwestliche Zone-Normalzeit",
            "daylight";
            "Mexiko Nordwestliche Zone-Sommerzeit";
        }
    }
    "Mexico_Pacific";
    {
        "long";
        {
            "generic";
            "Mexiko Pazifikzone-Zeit",
            "standard";
            "Mexiko Pazifikzone-Normalzeit",
            "daylight";
            "Mexiko Pazifikzone-Sommerzeit";
        }
    }
    "Mongolia";
    {
        "long";
        {

```

```

        "generic";
        "Ulaanbaatar-Zeit",
        "standard";
        "Ulaanbaatar-Normalzeit",
        "daylight";
        "Ulaanbaatar-Sommerzeit";
    }
}
"Moscow";
{
    "long";
    {
        "generic";
        "Moskauer Zeit",
        "standard";
        "Moskauer Normalzeit",
        "daylight";
        "Moskauer Sommerzeit";
    }
}
"Myanmar";
{
    "long";
    {
        "standard";
        "Myanmar-Zeit";
    }
}
"Nauru";
{
    "long";
    {
        "standard";
        "Nauru-Zeit";
    }
}
"Nepal";
{
    "long";
    {
        "standard";
        "Nepalesische Zeit";
    }
}
"New_Caledonia";
{
    "long";
    {
        "generic";
        "Neukaledonische Zeit",
        "standard";
        "Neukaledonische Normalzeit",
        "daylight";
        "Neukaledonische Sommerzeit";
    }
}
"New_Zealand";

```



```
{
  "long";
  {
    "generic";
    "Neuseeland-Zeit",
    "standard";
    "Neuseeland-Normalzeit",
    "daylight";
    "Neuseeland-Sommerzeit";
  }
}
"Newfoundland";
{
  "long";
  {
    "generic";
    "Neufundland-Zeit",
    "standard";
    "Neufundland-Normalzeit",
    "daylight";
    "Neufundland-Sommerzeit";
  }
}
"Niue";
{
  "long";
  {
    "standard";
    "Niue-Zeit";
  }
}
"Norfolk";
{
  "long";
  {
    "standard";
    "Norfolkinsel-Zeit";
  }
}
"Noronha";
{
  "long";
  {
    "generic";
    "Fernando de Noronha-Zeit",
    "standard";
    "Fernando de Noronha-Normalzeit",
    "daylight";
    "Fernando de Noronha-Sommerzeit";
  }
}
"North_Mariana";
{
  "long";
  {
    "standard";
    "Nördliche-Marianen-Zeit";
  }
}
```

```

    }
  }
  "Novosibirsk";
  {
    "long";
    {
      "generic";
      "Novosibirsk-Zeit",
      "standard";
      "Novosibirsk-Normalzeit",
      "daylight";
      "Novosibirsk-Sommerzeit";
    }
  }
  "Omsk";
  {
    "long";
    {
      "generic";
      "Omsk-Zeit",
      "standard";
      "Omsk-Normalzeit",
      "daylight";
      "Omsk-Sommerzeit";
    }
  }
  "Pakistan";
  {
    "long";
    {
      "generic";
      "Pakistanische Zeit",
      "standard";
      "Pakistanische Normalzeit",
      "daylight";
      "Pakistanische Sommerzeit";
    }
  }
  "Palau";
  {
    "long";
    {
      "standard";
      "Palau-Zeit";
    }
  }
  "Papua_New_Guinea";
  {
    "long";
    {
      "standard";
      "Papua-Neuguinea-Zeit";
    }
  }
  "Paraguay";
  {
    "long";

```

```

        {
            "generic";
            "Paraguayianische Zeit",
            "standard";
            "Paraguayianische Normalzeit",
            "daylight";
            "Paraguayianische Sommerzeit";
        }
    }
    "Peru";
    {
        "long";
        {
            "generic";
            "Peruanische Zeit",
            "standard";
            "Peruanische Normalzeit",
            "daylight";
            "Peruanische Sommerzeit";
        }
    }
    "Philippines";
    {
        "long";
        {
            "generic";
            "Philippinische Zeit",
            "standard";
            "Philippinische Normalzeit",
            "daylight";
            "Philippinische Sommerzeit";
        }
    }
    "Phoenix_Islands";
    {
        "long";
        {
            "standard";
            "Phoenixinseln-Zeit";
        }
    }
    "Pierre_Miquelon";
    {
        "long";
        {
            "generic";
            "Saint-Pierre-und-Miquelon-Zeit",
            "standard";
            "Saint-Pierre-und-Miquelon-Normalzeit",
            "daylight";
            "Saint-Pierre-und-Miquelon-Sommerzeit";
        }
    }
    "Pitcairn";
    {
        "long";
        {

```

```

        "standard";
        "Pitcairninseln-Zeit";
    }
}
"Ponape";
{
    "long";
    {
        "standard";
        "Ponape-Zeit";
    }
}
"Pyongyang";
{
    "long";
    {
        "standard";
        "Pjöngjang-Zeit";
    }
}
"Qyzylorda";
{
    "long";
    {
        "generic";
        "Quysylorda-Zeit",
        "standard";
        "Quysylorda-Normalzeit",
        "daylight";
        "Qysylorda-Sommerzeit";
    }
}
"Reunion";
{
    "long";
    {
        "standard";
        "Réunion-Zeit";
    }
}
"Rothera";
{
    "long";
    {
        "standard";
        "Rothera-Zeit";
    }
}
"Sakhalin";
{
    "long";
    {
        "generic";
        "Sachalin-Zeit",
        "standard";
        "Sachalin-Normalzeit",
        "daylight";
    }
}

```

```
        "Sachalin-Sommerzeit";
    }
}
"Samara";
{
    "long";
    {
        "generic";
        "Samara-Zeit",
        "standard";
        "Samara-Normalzeit",
        "daylight";
        "Samara-Sommerzeit";
    }
}
"Samoa";
{
    "long";
    {
        "generic";
        "Samoa-Zeit",
        "standard";
        "Samoa-Normalzeit",
        "daylight";
        "Samoa-Sommerzeit";
    }
}
"Seychelles";
{
    "long";
    {
        "standard";
        "Seychellen-Zeit";
    }
}
"Singapore";
{
    "long";
    {
        "standard";
        "Singapur-Zeit";
    }
}
"Solomon";
{
    "long";
    {
        "standard";
        "Salomoninseln-Zeit";
    }
}
"South_Georgia";
{
    "long";
    {
        "standard";
        "Südgeorgische Zeit";
    }
}
```

```
    }  
  }  
  "Suriname";  
  {  
    "long";  
    {  
      "standard";  
      "Suriname-Zeit";  
    }  
  }  
  "Syowa";  
  {  
    "long";  
    {  
      "standard";  
      "Syowa-Zeit";  
    }  
  }  
  "Tahiti";  
  {  
    "long";  
    {  
      "standard";  
      "Tahiti-Zeit";  
    }  
  }  
  "Taipei";  
  {  
    "long";  
    {  
      "generic";  
      "Taipeh-Zeit",  
        "standard";  
      "Taipeh-Normalzeit",  
        "daylight";  
      "Taipeh-Sommerzeit";  
    }  
  }  
  "Tajikistan";  
  {  
    "long";  
    {  
      "standard";  
      "Tadschikistan-Zeit";  
    }  
  }  
  "Tokelau";  
  {  
    "long";  
    {  
      "standard";  
      "Tokelau-Zeit";  
    }  
  }  
  "Tonga";  
  {  
    "long";
```

```

        {
            "generic";
            "Tonganische Zeit",
            "standard";
            "Tonganische Normalzeit",
            "daylight";
            "Tonganische Sommerzeit";
        }
    }
    "Truk";
    {
        "long";
        {
            "standard";
            "Chuuk-Zeit";
        }
    }
    "Turkmenistan";
    {
        "long";
        {
            "generic";
            "Turkmenistan-Zeit",
            "standard";
            "Turkmenistan-Normalzeit",
            "daylight";
            "Turkmenistan-Sommerzeit";
        }
    }
    "Tuvalu";
    {
        "long";
        {
            "standard";
            "Tuvalu-Zeit";
        }
    }
    "Uruguay";
    {
        "long";
        {
            "generic";
            "Uruguayische Zeit",
            "standard";
            "Uruguayische Normalzeit",
            "daylight";
            "Uruguayische Sommerzeit";
        }
    }
    "Uzbekistan";
    {
        "long";
        {
            "generic";
            "Uzbekistan-Zeit",
            "standard";
            "Uzbekistan-Normalzeit",

```

```

        "daylight";
        "Usbekistan-Sommerzeit";
    }
}
"Vanuatu";
{
    "long";
    {
        "generic";
        "Vanuatu-Zeit",
        "standard";
        "Vanuatu-Normalzeit",
        "daylight";
        "Vanuatu-Sommerzeit";
    }
}
"Venezuela";
{
    "long";
    {
        "standard";
        "Venezuela-Zeit";
    }
}
"Vladivostok";
{
    "long";
    {
        "generic";
        "Wladiwostok-Zeit",
        "standard";
        "Wladiwostok-Normalzeit",
        "daylight";
        "Wladiwostok-Sommerzeit";
    }
}
"Volgograd";
{
    "long";
    {
        "generic";
        "Wolgograd-Zeit",
        "standard";
        "Wolgograd-Normalzeit",
        "daylight";
        "Wolgograd-Sommerzeit";
    }
}
"Vostok";
{
    "long";
    {
        "standard";
        "Wostok-Zeit";
    }
}
"Wake";

```



```

        {
            "long";
            {
                "standard";
                "Wake-Insel-Zeit";
            }
        }
        "Wallis";
        {
            "long";
            {
                "standard";
                "Wallis-und-Futuna-Zeit";
            }
        }
        "Yakutsk";
        {
            "long";
            {
                "generic";
                "Jakutsk-Zeit",
                "standard";
                "Jakutsk-Normalzeit",
                "daylight";
                "Jakutsk-Sommerzeit";
            }
        }
        "Yekaterinburg";
        {
            "long";
            {
                "generic";
                "Jekaterinburg-Zeit",
                "standard";
                "Jekaterinburg-Normalzeit",
                "daylight";
                "Jekaterinburg-Sommerzeit";
            }
        }
    }
}

```

[Functional-component]

CA-GREGORIAN.JSON

```

{
    "main": {
        "de": {
            "identity": {
                "version": {
                    "_number": "$Revision: 12879 $",

```

```

    "_cldrVersion": "30.0.3"
  },
  "language": "de"
},
"dates": {
  "calendars": {
    "gregorian": {
      "months": {
        "format": {
          "abbreviated": {
            "1": "Jan.",
            "2": "Feb.",
            "3": "März",
            "4": "Apr.",
            "5": "Mai",
            "6": "Juni",
            "7": "Juli",
            "8": "Aug.",
            "9": "Sep.",
            "10": "Okt.",
            "11": "Nov.",
            "12": "Dez."
          },
          "narrow": {
            "1": "J",
            "2": "F",
            "3": "M",
            "4": "A",
            "5": "M",
            "6": "J",
            "7": "J",
            "8": "A",
            "9": "S",
            "10": "O",
            "11": "N",
            "12": "D"
          },
          "wide": {
            "1": "Januar",
            "2": "Februar",
            "3": "März",
            "4": "April",
            "5": "Mai",
            "6": "Juni",
            "7": "Juli",
            "8": "August",
            "9": "September",
            "10": "Oktober",
            "11": "November",
            "12": "Dezember"
          }
        }
      },
      "stand-alone": {
        "abbreviated": {
          "1": "Jan",
          "2": "Feb",
          "3": "Mär",

```

```

        "4": "Apr",
        "5": "Mai",
        "6": "Jun",
        "7": "Jul",
        "8": "Aug",
        "9": "Sep",
        "10": "Okt",
        "11": "Nov",
        "12": "Dez"
    },
    "narrow": {
        "1": "J",
        "2": "F",
        "3": "M",
        "4": "A",
        "5": "M",
        "6": "J",
        "7": "J",
        "8": "A",
        "9": "S",
        "10": "O",
        "11": "N",
        "12": "D"
    },
    "wide": {
        "1": "Januar",
        "2": "Februar",
        "3": "März",
        "4": "April",
        "5": "Mai",
        "6": "Juni",
        "7": "Juli",
        "8": "August",
        "9": "September",
        "10": "Oktober",
        "11": "November",
        "12": "Dezember"
    }
}
},
"days": {
    "format": {
        "abbreviated": {
            "sun": "So.",
            "mon": "Mo.",
            "tue": "Di.",
            "wed": "Mi.",
            "thu": "Do.",
            "fri": "Fr.",
            "sat": "Sa."
        },
        "narrow": {
            "sun": "S",
            "mon": "M",
            "tue": "D",
            "wed": "M",
            "thu": "D",

```

```

        "fri": "F",
        "sat": "S"
    },
    "short": {
        "sun": "So.",
        "mon": "Mo.",
        "tue": "Di.",
        "wed": "Mi.",
        "thu": "Do.",
        "fri": "Fr.",
        "sat": "Sa."
    },
    "wide": {
        "sun": "Sonntag",
        "mon": "Montag",
        "tue": "Dienstag",
        "wed": "Mittwoch",
        "thu": "Donnerstag",
        "fri": "Freitag",
        "sat": "Samstag"
    }
},
"stand-alone": {
    "abbreviated": {
        "sun": "So",
        "mon": "Mo",
        "tue": "Di",
        "wed": "Mi",
        "thu": "Do",
        "fri": "Fr",
        "sat": "Sa"
    },
    "narrow": {
        "sun": "S",
        "mon": "M",
        "tue": "D",
        "wed": "M",
        "thu": "D",
        "fri": "F",
        "sat": "S"
    },
    "short": {
        "sun": "So.",
        "mon": "Mo.",
        "tue": "Di.",
        "wed": "Mi.",
        "thu": "Do.",
        "fri": "Fr.",
        "sat": "Sa."
    },
    "wide": {
        "sun": "Sonntag",
        "mon": "Montag",
        "tue": "Dienstag",
        "wed": "Mittwoch",
        "thu": "Donnerstag",
        "fri": "Freitag",

```

```

        "sat": "Samstag"
      }
    },
    "quarters": {
      "format": {
        "abbreviated": {
          "1": "Q1",
          "2": "Q2",
          "3": "Q3",
          "4": "Q4"
        },
        "narrow": {
          "1": "1",
          "2": "2",
          "3": "3",
          "4": "4"
        },
        "wide": {
          "1": "1. Quartal",
          "2": "2. Quartal",
          "3": "3. Quartal",
          "4": "4. Quartal"
        }
      },
      "stand-alone": {
        "abbreviated": {
          "1": "Q1",
          "2": "Q2",
          "3": "Q3",
          "4": "Q4"
        },
        "narrow": {
          "1": "1",
          "2": "2",
          "3": "3",
          "4": "4"
        },
        "wide": {
          "1": "1. Quartal",
          "2": "2. Quartal",
          "3": "3. Quartal",
          "4": "4. Quartal"
        }
      }
    },
    "dayPeriods": {
      "format": {
        "abbreviated": {
          "midnight": "Mitternacht",
          "am": "vorm.",
          "pm": "nachm.",
          "morning1": "morgens",
          "morning2": "vormittags",
          "afternoon1": "mittags",
          "afternoon2": "nachmittags",
          "evening1": "abends",

```

```

        "night1": "nachts"
    },
    "narrow": {
        "midnight": "Mitternacht",
        "am": "vm.",
        "pm": "nm.",
        "morning1": "morgens",
        "morning2": "vormittags",
        "afternoon1": "mittags",
        "afternoon2": "nachmittags",
        "evening1": "abends",
        "night1": "nachts"
    },
    "wide": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "morgens",
        "morning2": "vormittags",
        "afternoon1": "mittags",
        "afternoon2": "nachmittags",
        "evening1": "abends",
        "night1": "nachts"
    }
},
"stand-alone": {
    "abbreviated": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "Morgen",
        "morning2": "Vormittag",
        "afternoon1": "Mittag",
        "afternoon2": "Nachmittag",
        "evening1": "Abend",
        "night1": "Nacht"
    },
    "narrow": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "Morgen",
        "morning2": "Vormittag",
        "afternoon1": "Mittag",
        "afternoon2": "Nachmittag",
        "evening1": "Abend",
        "night1": "Nacht"
    },
    "wide": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "Morgen",
        "morning2": "Vormittag",
        "afternoon1": "Mittag",
        "afternoon2": "Nachmittag",
        "evening1": "Abend",

```

```

        "night1": "Nacht"
    }
}
},
"eras": {
    "eraNames": {
        "0": "v. Chr.",
        "0-alt-variant": "vor unserer Zeitrechnung",
        "1": "n. Chr.",
        "1-alt-variant": "unserer Zeitrechnung"
    },
    "eraAbbr": {
        "0": "v. Chr.",
        "0-alt-variant": "v. u. Z.",
        "1": "n. Chr.",
        "1-alt-variant": "u. Z."
    },
    "eraNarrow": {
        "0": "v. Chr.",
        "0-alt-variant": "v. u. Z.",
        "1": "n. Chr.",
        "1-alt-variant": "u. Z."
    }
},
"dateFormats": {
    "full": "EEEE, d. MMMM y",
    "long": "d. MMMM y",
    "medium": "dd.MM.y",
    "short": "dd.MM.yy"
},
"timeFormats": {
    "full": "HH:mm:ss zzzz",
    "long": "HH:mm:ss z",
    "medium": "HH:mm:ss",
    "short": "HH:mm"
},
"dateTimeFormats": {
    "full": "{1} 'um' {0}",
    "long": "{1} 'um' {0}",
    "medium": "{1}, {0}",
    "short": "{1}, {0}",
    "availableFormats": {
        "d": "d",
        "E": "ccc",
        "Ed": "E, d.",
        "Ehm": "E h:mm a",
        "EHm": "E, HH:mm",
        "Ehms": "E, h:mm:ss a",
        "EHms": "E, HH:mm:ss",
        "Gy": "y G",
        "GyMMM": "MMM y G",
        "GyMMMd": "d. MMM y G",
        "GyMMMEd": "E, d. MMM y G",
        "h": "h 'Uhr' a",
        "H": "HH 'Uhr'",
        "hm": "h:mm a",
        "Hm": "HH:mm",
    }
}

```

```

    "hms": "h:mm:ss a",
    "Hms": "HH:mm:ss",
    "hmsv": "h:mm:ss a v",
    "Hmsv": "HH:mm:ss v",
    "hmv": "h:mm a v",
    "Hmv": "HH:mm v",
    "M": "L",
    "Md": "d.M.",
    "MEd": "E, d.M.",
    "MMd": "d.MM.",
    "MMdd": "dd.MM.",
    "MMM": "LLL",
    "MMMd": "d. MMM",
    "MMMEd": "E, d. MMM",
    "MMMMd": "d. MMMM",
    "MMMMEd": "E, d. MMMM",
    "MMMMW": "'Woche' W 'im' MMM",
    "MMMMW": "'Woche' W 'im' MMM",
    "ms": "mm:ss",
    "y": "y",
    "yM": "M.y",
    "yMd": "d.M.y",
    "yMEd": "E, d.M.y",
    "yMM": "MM.y",
    "yMMdd": "dd.MM.y",
    "yMMM": "MMM y",
    "yMMMd": "d. MMM y",
    "yMMMEd": "E, d. MMM y",
    "yMMMM": "MMMM y",
    "yQQQ": "QQQ y",
    "yQQQQ": "QQQQ y",
    "yw": "'Woche' w 'des' 'Jahres' y",
    "yw": "'Woche' w 'des' 'Jahres' y"
  },
  "appendItems": {
    "Day": "{0} ({2}: {1})",
    "Day-Of-Week": "{0} {1}",
    "Era": "{1} {0}",
    "Hour": "{0} ({2}: {1})",
    "Minute": "{0} ({2}: {1})",
    "Month": "{0} ({2}: {1})",
    "Quarter": "{0} ({2}: {1})",
    "Second": "{0} ({2}: {1})",
    "Timezone": "{0} {1}",
    "Week": "{0} ({2}: {1})",
    "Year": "{1} {0}"
  },
  "intervalFormats": {
    "intervalFormatFallback": "{0} - {1}",
    "d": {
      "d": "d.-d."
    },
    "h": {
      "a": "h 'Uhr' a - h 'Uhr' a",
      "h": "h - h 'Uhr' a"
    },
    "H": {

```



```

    "H": "HH-HH 'Uhr'",
  },
  "hm": {
    "a": "h:mm a - h:mm a",
    "h": "h:mm-h:mm a",
    "m": "h:mm-h:mm a"
  },
  "Hm": {
    "H": "HH:mm-HH:mm 'Uhr'",
    "m": "HH:mm-HH:mm 'Uhr'"
  },
  "hmv": {
    "a": "h:mm a - h:mm a v",
    "h": "h:mm-h:mm a v",
    "m": "h:mm-h:mm a v"
  },
  "Hmv": {
    "H": "HH:mm-HH:mm 'Uhr' v",
    "m": "HH:mm-HH:mm 'Uhr' v"
  },
  "hv": {
    "a": "h a - h a v",
    "h": "h-h a v"
  },
  "Hv": {
    "H": "HH-HH 'Uhr' v"
  },
  "M": {
    "M": "M.-M."
  },
  "Md": {
    "d": "dd.MM. - dd.MM.",
    "M": "dd.MM. - dd.MM."
  },
  "MEd": {
    "d": "E, dd.MM. - E, dd.MM.",
    "M": "E, dd.MM. - E, dd.MM."
  },
  "MMM": {
    "M": "MMM-MMM"
  },
  "MMMd": {
    "d": "d.-d. MMM",
    "M": "d. MMM - d. MMM"
  },
  "MMMED": {
    "d": "E, d. - E, d. MMM",
    "M": "E, d. MMM - E, d. MMM"
  },
  "MMMM": {
    "M": "LLLL-LLLL"
  },
  "Y": {
    "Y": "Y-Y"
  },
  "YM": {
    "M": "MM.Y - MM.Y",

```

```
"y": "MM.y - MM.y"
},
"yMd": {
    "d": "dd.MM.y - dd.MM.y",
    "M": "dd.MM.y - dd.MM.y",
    "Y": "dd.MM.y - dd.MM.Y"
},
"yMEd": {
    "d": "E, dd.MM.y - E, dd.MM.y",
    "M": "E, dd.MM.y - E, dd.MM.y",
    "Y": "E, dd.MM.y - E, dd.MM.Y"
},
"yMMM": {
    "M": "MMM-MMM y",
    "Y": "MMM Y - MMM Y"
},
"yMMMd": {
    "d": "d.-d. MMM y",
    "M": "d. MMM - d. MMM y",
    "Y": "d. MMM Y - d. MMM Y"
},
"yMMMED": {
    "d": "E, d. - E, d. MMM y",
    "M": "E, d. MMM - E, d. MMM y",
    "Y": "E, d. MMM Y - E, d. MMM Y"
},
"yMMMM": {
    "M": "MMMM-MMMM y",
    "Y": "MMMM Y - MMMM Y"
}
}
}
}
}
```

CA-GREGORIAN.JSX

```
{
  "main";
  {
    "de";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 12879 $" ,
            "_cldrVersion";
            "30.0.3";
        }
        "language";
      }
    }
  }
}
```

```

        "de";
    }
    "dates";
    {
        "calendars";
        {
            "gregorian";
            {
                "months";
                {
                    "format";
                    {
                        "abbreviated";
                        {
                            "1";
                            "Jan.",
                                "2";
                            "Feb.",
                                "3";
                            "März",
                                "4";
                            "Apr.",
                                "5";
                            "Mai",
                                "6";
                            "Juni",
                                "7";
                            "Juli",
                                "8";
                            "Aug.",
                                "9";
                            "Sep.",
                                "10";
                            "Okt.",
                                "11";
                            "Nov.",
                                "12";
                            "Dez.";
                        }
                        "narrow";
                        {
                            "1";
                            "J",
                                "2";
                            "F",
                                "3";
                            "M",
                                "4";
                            "A",
                                "5";
                            "M",
                                "6";
                            "J",
                                "7";
                            "J",
                                "8";
                            "A",

```

```

        "9";
        "S",
        "10";
        "O",
        "11";
        "N",
        "12";
        "D";
    }
    "wide";
    {
        "1";
        "Januar",
        "2";
        "Februar",
        "3";
        "März",
        "4";
        "April",
        "5";
        "Mai",
        "6";
        "Juni",
        "7";
        "Juli",
        "8";
        "August",
        "9";
        "September",
        "10";
        "Oktober",
        "11";
        "November",
        "12";
        "Dezember";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
        "Jan",
        "2";
        "Feb",
        "3";
        "Mär",
        "4";
        "Apr",
        "5";
        "Mai",
        "6";
        "Jun",
        "7";
        "Jul",
        "8";
        "Aug",

```

```

        "9";
        "Sep",
        "10";
        "Okt",
        "11";
        "Nov",
        "12";
        "Dez";
    }
    "narrow";
    {
        "1";
        "J",
        "2";
        "F",
        "3";
        "M",
        "4";
        "A",
        "5";
        "M",
        "6";
        "J",
        "7";
        "J",
        "8";
        "A",
        "9";
        "S",
        "10";
        "O",
        "11";
        "N",
        "12";
        "D";
    }
    "wide";
    {
        "1";
        "Januar",
        "2";
        "Februar",
        "3";
        "März",
        "4";
        "April",
        "5";
        "Mai",
        "6";
        "Juni",
        "7";
        "Juli",
        "8";
        "August",
        "9";
        "September",
        "10";
    }

```

```

        "Oktober",
        "11";
        "November",
        "12";
        "Dezember";
    }
}
}
"days";
{
    "format";
    {
        "abbreviated";
        {
            "sun";
            "So.",
            "mon";
            "Mo.",
            "tue";
            "Di.",
            "wed";
            "Mi.",
            "thu";
            "Do.",
            "fri";
            "Fr.",
            "sat";
            "Sa.";
        }
        "narrow";
        {
            "sun";
            "S",
            "mon";
            "M",
            "tue";
            "D",
            "wed";
            "M",
            "thu";
            "D",
            "fri";
            "F",
            "sat";
            "S";
        }
        "short";
        {
            "sun";
            "So.",
            "mon";
            "Mo.",
            "tue";
            "Di.",
            "wed";
            "Mi.",
            "thu";

```

```
        "Do.",
        "fri";
        "Fr.",
        "sat";
        "Sa.";
    }
    "wide";
    {
        "sun";
        "Sonntag",
        "mon";
        "Montag",
        "tue";
        "Dienstag",
        "wed";
        "Mittwoch",
        "thu";
        "Donnerstag",
        "fri";
        "Freitag",
        "sat";
        "Samstag";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "sun";
        "So",
        "mon";
        "Mo",
        "tue";
        "Di",
        "wed";
        "Mi",
        "thu";
        "Do",
        "fri";
        "Fr",
        "sat";
        "Sa";
    }
    "narrow";
    {
        "sun";
        "S",
        "mon";
        "M",
        "tue";
        "D",
        "wed";
        "M",
        "thu";
        "D",
        "fri";
        "F",
```

```

        "sat";
        "S";
    }
    "short";
    {
        "sun";
        "So.",
        "mon";
        "Mo.",
        "tue";
        "Di.",
        "wed";
        "Mi.",
        "thu";
        "Do.",
        "fri";
        "Fr.",
        "sat";
        "Sa.";
    }
    "wide";
    {
        "sun";
        "Sonntag",
        "mon";
        "Montag",
        "tue";
        "Dienstag",
        "wed";
        "Mittwoch",
        "thu";
        "Donnerstag",
        "fri";
        "Freitag",
        "sat";
        "Samstag";
    }
}
"quarters";
{
    "format";
    {
        "abbreviated";
        {
            "1";
            "Q1",
            "2";
            "Q2",
            "3";
            "Q3",
            "4";
            "Q4";
        }
        "narrow";
        {
            "1";

```



```

        "1",
        "2";
        "2",
        "3";
        "3",
        "4";
        "4";
    }
    "wide";
    {
        "1";
        "1. Quartal",
        "2";
        "2. Quartal",
        "3";
        "3. Quartal",
        "4";
        "4. Quartal";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
        "Q1",
        "2";
        "Q2",
        "3";
        "Q3",
        "4";
        "Q4";
    }
    "narrow";
    {
        "1";
        "1",
        "2";
        "2",
        "3";
        "3",
        "4";
        "4";
    }
    "wide";
    {
        "1";
        "1. Quartal",
        "2";
        "2. Quartal",
        "3";
        "3. Quartal",
        "4";
        "4. Quartal";
    }
}
}
}

```

```

    "dayPeriods";
    {
      "format";
      {
        "abbreviated";
        {
          "midnight";
          "Mitternacht",
          "am";
          "vorm.",
          "pm";
          "nachm.",
          "morning1";
          "morgens",
          "morning2";
          "vormittags",
          "afternoon1";
          "mittags",
          "afternoon2";
          "nachmittags",
          "evening1";
          "abends",
          "night1";
          "nachts";
        }
        "narrow";
        {
          "midnight";
          "Mitternacht",
          "am";
          "vm.",
          "pm";
          "nm.",
          "morning1";
          "morgens",
          "morning2";
          "vormittags",
          "afternoon1";
          "mittags",
          "afternoon2";
          "nachmittags",
          "evening1";
          "abends",
          "night1";
          "nachts";
        }
        "wide";
        {
          "midnight";
          "Mitternacht",
          "am";
          "vorm.",
          "pm";
          "nachm.",
          "morning1";
          "morgens",
          "morning2";
        }
      }
    }
  
```

```

        "vormittags",
        "afternoon1";
    "mittags",
        "afternoon2";
    "nachmittags",
        "evening1";
    "abends",
        "night1";
    "nachts";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "midnight";
        "Mitternacht",
            "am";
        "vorm.",
            "pm";
        "nachm.",
            "morning1";
        "Morgen",
            "morning2";
        "Vormittag",
            "afternoon1";
        "Mittag",
            "afternoon2";
        "Nachmittag",
            "evening1";
        "Abend",
            "night1";
        "Nacht";
    }
    "narrow";
    {
        "midnight";
        "Mitternacht",
            "am";
        "vorm.",
            "pm";
        "nachm.",
            "morning1";
        "Morgen",
            "morning2";
        "Vormittag",
            "afternoon1";
        "Mittag",
            "afternoon2";
        "Nachmittag",
            "evening1";
        "Abend",
            "night1";
        "Nacht";
    }
    "wide";
    {

```

```

        "midnight";
        "Mitternacht",
            "am";
        "vorm.",
            "pm";
        "nachm.",
            "morning1";
        "Morgen",
            "morning2";
        "Vormittag",
            "afternoon1";
        "Mittag",
            "afternoon2";
        "Nachmittag",
            "evening1";
        "Abend",
            "night1";
        "Nacht";
    }
}
"eras";
{
    "eraNames";
    {
        "0";
        "v. Chr.",
            "0-alt-variant";
        "vor unserer Zeitrechnung",
            "1";
        "n. Chr.",
            "1-alt-variant";
        "unserer Zeitrechnung";
    }
    "eraAbbr";
    {
        "0";
        "v. Chr.",
            "0-alt-variant";
        "v. u. Z.",
            "1";
        "n. Chr.",
            "1-alt-variant";
        "u. Z.";
    }
    "eraNarrow";
    {
        "0";
        "v. Chr.",
            "0-alt-variant";
        "v. u. Z.",
            "1";
        "n. Chr.",
            "1-alt-variant";
        "u. Z.";
    }
}

```

```

    "dateFormats";
    {
        "full";
        "EEEE, d. MMMM y",
        "long";
        "d. MMMM y",
        "medium";
        "dd.MM.y",
        "short";
        "dd.MM.yy";
    }
    "timeFormats";
    {
        "full";
        "HH:mm:ss zzzz",
        "long";
        "HH:mm:ss z",
        "medium";
        "HH:mm:ss",
        "short";
        "HH:mm";
    }
    "dateTimeFormats";
    {
        "full";
        "{1} 'um' {0}",
        "long";
        "{1} 'um' {0}",
        "medium";
        "{1}, {0}",
        "short";
        "{1}, {0}",
        "availableFormats";
        {
            "d";
            "d",
            "E";
            "ccc",
            "Ed";
            "E, d.",
            "Ehm";
            "E h:mm a",
            "EHm";
            "E, HH:mm",
            "Ehms";
            "E, h:mm:ss a",
            "EHms";
            "E, HH:mm:ss",
            "Gy";
            "y G",
            "GyMMM";
            "MMM y G",
            "GyMMMd";
            "d. MMM y G",
            "GyMMMED";
            "E, d. MMM y G",
            "h";
        }
    }

```

```

    "h 'Uhr' a",
        "H";
    "HH 'Uhr' ",
        "hm";
    "h:mm a",
        "Hm";
    "HH:mm",
        "hms";
    "h:mm:ss a",
        "Hms";
    "HH:mm:ss",
        "hmsv";
    "h:mm:ss a v",
        "Hmsv";
    "HH:mm:ss v",
        "hmv";
    "h:mm a v",
        "Hmv";
    "HH:mm v",
        "M";
    "L",
        "Md";
    "d.M.",
        "MEd";
    "E, d.M.",
        "MMd";
    "d.MM.",
        "MMdd";
    "dd.MM.",
        "MMM";
    "LLL",
        "MMMd";
    "d. MMM",
        "MMMEd";
    "E, d. MMM",
        "MMMMd";
    "d. MMMM",
        "MMMMEd";
    "E, d. MMMM",
        "MMMMW";
    "'Woche' W 'im' MMM",
        "MMMMW";
    "'Woche' W 'im' MMM",
        "ms";
    "mm:ss",
        "y";
    "Y",
        "yM";
    "M.y",
        "yMd";
    "d.M.y",
        "yMEd";
    "E, d.M.y",
        "yMM";
    "MM.y",
        "yMMdd";
    "dd.MM.y",

```

```

        "yMMM";
        "MMM y",
        "yMMMd";
        "d. MMM y",
        "yMMMEd";
        "E, d. MMM y",
        "yMMMM";
        "MMMM y",
        "yQQQ";
        "QQQ y",
        "yQQQQ";
        "QQQQ y",
        "yw";
        "'Woche' w 'des' 'Jahres' y",
        "yw";
        "'Woche' w 'des' 'Jahres' y";
    }
    "appendItems";
    {
        "Day";
        "{0} ({2}: {1})",
        "Day-Of-Week";
        "{0} {1}",
        "Era";
        "{1} {0}",
        "Hour";
        "{0} ({2}: {1})",
        "Minute";
        "{0} ({2}: {1})",
        "Month";
        "{0} ({2}: {1})",
        "Quarter";
        "{0} ({2}: {1})",
        "Second";
        "{0} ({2}: {1})",
        "Timezone";
        "{0} {1}",
        "Week";
        "{0} ({2}: {1})",
        "Year";
        "{1} {0}";
    }
    "intervalFormats";
    {
        "intervalFormatFallback";
        "{0} - {1}",
        "d";
        {
            "d";
            "d.-d.";
        }
        "h";
        {
            "a";
            "h 'Uhr' a - h 'Uhr' a",
            "h";
            "h - h 'Uhr' a";
        }
    }

```

```

    }
    "H";
    {
        "H";
        "HH-HH 'Uhr'";
    }
    "hm";
    {
        "a";
        "h:mm a - h:mm a",
        "h";
        "h:mm-h:mm a",
        "m";
        "h:mm-h:mm a";
    }
    "Hm";
    {
        "H";
        "HH:mm-HH:mm 'Uhr'",
        "m";
        "HH:mm-HH:mm 'Uhr'";
    }
    "hmv";
    {
        "a";
        "h:mm a - h:mm a v",
        "h";
        "h:mm-h:mm a v",
        "m";
        "h:mm-h:mm a v";
    }
    "Hmv";
    {
        "H";
        "HH:mm-HH:mm 'Uhr' v",
        "m";
        "HH:mm-HH:mm 'Uhr' v";
    }
    "hv";
    {
        "a";
        "h a - h a v",
        "h";
        "h-h a v";
    }
    "Hv";
    {
        "H";
        "HH-HH 'Uhr' v";
    }
    "M";
    {
        "M";
        "M.-M.";
    }
    "Md";
    {

```



```

        "d";
        "dd.MM. - dd.MM.",
        "M";
        "dd.MM. - dd.MM.";
    }
    "MEd";
    {
        "d";
        "E, dd.MM. - E, dd.MM.",
        "M";
        "E, dd.MM. - E, dd.MM.";
    }
    "MMM";
    {
        "M";
        "MMM-MMM";
    }
    "MMMd";
    {
        "d";
        "d.-d. MMM",
        "M";
        "d. MMM - d. MMM";
    }
    "MMMed";
    {
        "d";
        "E, d. - E, d. MMM",
        "M";
        "E, d. MMM - E, d. MMM";
    }
    "MMMM";
    {
        "M";
        "LLLL-LLLL";
    }
    "Y";
    {
        "Y";
        "Y-Y";
    }
    "YM";
    {
        "M";
        "MM.y - MM.y",
        "Y";
        "MM.y - MM.y";
    }
    "yMd";
    {
        "d";
        "dd.MM.y - dd.MM.y",
        "M";
        "dd.MM.y - dd.MM.y",
        "Y";
        "dd.MM.y - dd.MM.y";
    }
}

```

```

    "yMEd";
    {
        "d";
        "E, dd.MM.y - E, dd.MM.y",
        "M";
        "E, dd.MM.y - E, dd.MM.y",
        "y";
        "E, dd.MM.y - E, dd.MM.y";
    }
    "yMMM";
    {
        "M";
        "MMM-MMM y",
        "y";
        "MMM y - MMM y";
    }
    "yMMMd";
    {
        "d";
        "d.-d. MMM y",
        "M";
        "d. MMM - d. MMM y",
        "y";
        "d. MMM y - d. MMM y";
    }
    "yMMMEd";
    {
        "d";
        "E, d. - E, d. MMM y",
        "M";
        "E, d. MMM - E, d. MMM y",
        "y";
        "E, d. MMM y - E, d. MMM y";
    }
    "yMMMM";
    {
        "M";
        "MMMM-MMMM y",
        "y";
        "MMMM y - MMMM y";
    }
}
}
}
}
}
}
}
}
}
}

```

CURRENCIES.JSON

```
{
  "main": {
    "de": {
      "identity": {
```

```

    "version": {
      "_number": "$Revision: 13259 $",
      "_cldrVersion": "31"
    },
    "language": "de"
  },
  "numbers": {
    "currencies": {
      "ADP": {
        "displayName": "Andorranische Pesete",
        "displayName-count-one": "Andorranische Pesete",
        "displayName-count-other": "Andorranische Peseten",
        "symbol": "ADP"
      },
      "AED": {
        "displayName": "VAE-Dirham",
        "displayName-count-one": "VAE-Dirham",
        "displayName-count-other": "VAE-Dirham",
        "symbol": "AED"
      },
      "AFA": {
        "displayName": "Afghanische Afghani (1927-2002)",
        "displayName-count-one": "Afghanische Afghani (1927-2002)",
        "displayName-count-other": "Afghanische Afghani (1927-2002)",
        "symbol": "AFA"
      },
      "AFN": {
        "displayName": "Afghanischer Afghani",
        "displayName-count-one": "Afghanischer Afghani",
        "displayName-count-other": "Afghanische Afghani",
        "symbol": "AFN"
      },
      "ALK": {
        "displayName": "Albanischer Lek (1946-1965)",
        "displayName-count-one": "Albanischer Lek (1946-1965)",
        "displayName-count-other": "Albanische Lek (1946-1965)"
      },
      "ALL": {
        "displayName": "Albanischer Lek",
        "displayName-count-one": "Albanischer Lek",
        "displayName-count-other": "Albanische Lek",
        "symbol": "ALL"
      },
      "AMD": {
        "displayName": "Armenischer Dram",
        "displayName-count-one": "Armenischer Dram",
        "displayName-count-other": "Armenische Dram",
        "symbol": "AMD"
      },
      "ANG": {
        "displayName": "Niederländische-Antillen-Gulden",
        "displayName-count-one": "Niederländische-Antillen-Gulden",
        "displayName-count-other": "Niederländische-Antillen-Gulden",
        "symbol": "ANG"
      },
      "AOA": {
        "displayName": "Angolanischer Kwanza",

```

```

        "displayName-count-one": "Angolanischer Kwanza",
        "displayName-count-other": "Angolanische Kwanza",
        "symbol": "AOA",
        "symbol-alt-narrow": "Kz"
    },
    "AOK": {
        "displayName": "Angolanischer Kwanza (1977-1990)",
        "displayName-count-one": "Angolanischer Kwanza (1977-1990)",
        "displayName-count-other": "Angolanische Kwanza (1977-1990)",
        "symbol": "AOK"
    },
    "AON": {
        "displayName": "Angolanischer Neuer Kwanza (1990-2000)",
        "displayName-count-one": "Angolanischer Neuer Kwanza (1990-
2000)",
        "displayName-count-other": "Angolanische Neue Kwanza (1990-
2000)",
        "symbol": "AON"
    },
    "AOR": {
        "displayName": "Angolanischer Kwanza Reajustado (1995-1999)",
        "displayName-count-one": "Angolanischer Kwanza Reajustado (1995-
1999)",
        "displayName-count-other": "Angolanische Kwanza Reajustado
(1995-1999)",
        "symbol": "AOR"
    },
    "ARA": {
        "displayName": "Argentinischer Austral",
        "displayName-count-one": "Argentinischer Austral",
        "displayName-count-other": "Argentinische Austral",
        "symbol": "ARA"
    },
    "ARL": {
        "displayName": "Argentinischer Peso Ley (1970-1983)",
        "displayName-count-one": "Argentinischer Peso Ley (1970-1983)",
        "displayName-count-other": "Argentinische Pesos Ley (1970-
1983)",
        "symbol": "ARL"
    },
    "ARM": {
        "displayName": "Argentinischer Peso (1881-1970)",
        "displayName-count-one": "Argentinischer Peso (1881-1970)",
        "displayName-count-other": "Argentinische Pesos (1881-1970)",
        "symbol": "ARM"
    },
    "ARP": {
        "displayName": "Argentinischer Peso (1983-1985)",
        "displayName-count-one": "Argentinischer Peso (1983-1985)",
        "displayName-count-other": "Argentinische Peso (1983-1985)",
        "symbol": "ARP"
    },
    "ARS": {
        "displayName": "Argentinischer Peso",
        "displayName-count-one": "Argentinischer Peso",
        "displayName-count-other": "Argentinische Pesos",
        "symbol": "ARS",

```

```

        "symbol-alt-narrow": "$"
    },
    "ATS": {
        "displayName": "Österreichischer Schilling",
        "displayName-count-one": "Österreichischer Schilling",
        "displayName-count-other": "Österreichische Schilling",
        "symbol": "öS"
    },
    "AUD": {
        "displayName": "Australischer Dollar",
        "displayName-count-one": "Australischer Dollar",
        "displayName-count-other": "Australische Dollar",
        "symbol": "AU$",
        "symbol-alt-narrow": "$"
    },
    "AWG": {
        "displayName": "Aruba-Florin",
        "displayName-count-one": "Aruba-Florin",
        "displayName-count-other": "Aruba-Florin",
        "symbol": "AWG"
    },
    "AZM": {
        "displayName": "Aserbaidtschan-Manat (1993-2006)",
        "displayName-count-one": "Aserbaidtschan-Manat (1993-2006)",
        "displayName-count-other": "Aserbaidtschan-Manat (1993-2006)",
        "symbol": "AZM"
    },
    "AZN": {
        "displayName": "Aserbaidtschan-Manat",
        "displayName-count-one": "Aserbaidtschan-Manat",
        "displayName-count-other": "Aserbaidtschan-Manat",
        "symbol": "AZN"
    },
    "BAD": {
        "displayName": "Bosnien und Herzegowina Dinar (1992-1994)",
        "displayName-count-one": "Bosnien und Herzegowina Dinar (1992-1994)",
        "displayName-count-other": "Bosnien und Herzegowina Dinar (1992-1994)",
        "symbol": "BAD"
    },
    "BAM": {
        "displayName": "Bosnien und Herzegowina Konvertierbare Mark",
        "displayName-count-one": "Bosnien und Herzegowina Konvertierbare Mark",
        "displayName-count-other": "Bosnien und Herzegowina Konvertierbare Mark",
        "symbol": "BAM",
        "symbol-alt-narrow": "KM"
    },
    "BAN": {
        "displayName": "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
        "displayName-count-one": "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
        "displayName-count-other": "Bosnien und Herzegowina Neue Dinar (1994-1997)",

```

```

        "symbol": "BAN"
    },
    "BBD": {
        "displayName": "Barbados-Dollar",
        "displayName-count-one": "Barbados-Dollar",
        "displayName-count-other": "Barbados-Dollar",
        "symbol": "BBD",
        "symbol-alt-narrow": "$"
    },
    "BDT": {
        "displayName": "Bangladesch-Taka",
        "displayName-count-one": "Bangladesch-Taka",
        "displayName-count-other": "Bangladesch-Taka",
        "symbol": "BDT",
        "symbol-alt-narrow": "ট"
    },
    "BEC": {
        "displayName": "Belgischer Franc (konvertibel)",
        "displayName-count-one": "Belgischer Franc (konvertibel)",
        "displayName-count-other": "Belgische Franc (konvertibel)",
        "symbol": "BEC"
    },
    "BEF": {
        "displayName": "Belgischer Franc",
        "displayName-count-one": "Belgischer Franc",
        "displayName-count-other": "Belgische Franc",
        "symbol": "BEF"
    },
    "BEL": {
        "displayName": "Belgischer Finanz-Franc",
        "displayName-count-one": "Belgischer Finanz-Franc",
        "displayName-count-other": "Belgische Finanz-Franc",
        "symbol": "BEL"
    },
    "BGL": {
        "displayName": "Bulgarische Lew (1962-1999)",
        "displayName-count-one": "Bulgarische Lew (1962-1999)",
        "displayName-count-other": "Bulgarische Lew (1962-1999)",
        "symbol": "BGL"
    },
    "BGM": {
        "displayName": "Bulgarischer Lew (1952-1962)",
        "displayName-count-one": "Bulgarischer Lew (1952-1962)",
        "displayName-count-other": "Bulgarische Lew (1952-1962)",
        "symbol": "BGK"
    },
    "BGN": {
        "displayName": "Bulgarischer Lew",
        "displayName-count-one": "Bulgarischer Lew",
        "displayName-count-other": "Bulgarische Lew",
        "symbol": "BGN"
    },
    "BGO": {
        "displayName": "Bulgarischer Lew (1879-1952)",
        "displayName-count-one": "Bulgarischer Lew (1879-1952)",
        "displayName-count-other": "Bulgarische Lew (1879-1952)",

```

```

        "symbol": "BGJ"
    },
    "BHD": {
        "displayName": "Bahrain-Dinar",
        "displayName-count-one": "Bahrain-Dinar",
        "displayName-count-other": "Bahrain-Dinar",
        "symbol": "BHD"
    },
    "BIF": {
        "displayName": "Burundi-Franc",
        "displayName-count-one": "Burundi-Franc",
        "displayName-count-other": "Burundi-Franks",
        "symbol": "BIF"
    },
    "BMD": {
        "displayName": "Bermuda-Dollar",
        "displayName-count-one": "Bermuda-Dollar",
        "displayName-count-other": "Bermuda-Dollar",
        "symbol": "BMD",
        "symbol-alt-narrow": "$"
    },
    "BND": {
        "displayName": "Brunei-Dollar",
        "displayName-count-one": "Brunei-Dollar",
        "displayName-count-other": "Brunei-Dollar",
        "symbol": "BND",
        "symbol-alt-narrow": "$"
    },
    "BOB": {
        "displayName": "Bolivanischer Boliviano",
        "displayName-count-one": "Bolivanischer Boliviano",
        "displayName-count-other": "Bolivianische Bolivianos",
        "symbol": "BOB",
        "symbol-alt-narrow": "Bs"
    },
    "BOL": {
        "displayName": "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-one": "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-other": "Bolivianische Bolivianos (1863-
1963)",
        "symbol": "BOL"
    },
    "BOP": {
        "displayName": "Bolivianischer Peso",
        "displayName-count-one": "Bolivianischer Peso",
        "displayName-count-other": "Bolivianische Peso",
        "symbol": "BOP"
    },
    "BOV": {
        "displayName": "Boliviansiche Mvdol",
        "displayName-count-one": "Boliviansiche Mvdol",
        "displayName-count-other": "Bolivianische Mvdol",
        "symbol": "BOV"
    },
    "BRB": {
        "displayName": "Brasilianischer Cruzeiro Novo (1967-1986)",

```

```

    "displayName-count-one": "Brasilianischer Cruzeiro Novo (1967-
1986) ",
    "displayName-count-other": "Brasilianische Cruzeiro Novo (1967-
1986) ",
    "symbol": "BRB"
  },
  "BRC": {
    "displayName": "Brasilianischer Cruzado (1986-1989)",
    "displayName-count-one": "Brasilianischer Cruzado (1986-1989)",
    "displayName-count-other": "Brasilianische Cruzado (1986-1989)",
    "symbol": "BRC"
  },
  "BRE": {
    "displayName": "Brasilianischer Cruzeiro (1990-1993)",
    "displayName-count-one": "Brasilianischer Cruzeiro (1990-1993)",
    "displayName-count-other": "Brasilianische Cruzeiro (1990-
1993) ",
    "symbol": "BRE"
  },
  "BRL": {
    "displayName": "Brasilianischer Real",
    "displayName-count-one": "Brasilianischer Real",
    "displayName-count-other": "Brasilianische Real",
    "symbol": "R$",
    "symbol-alt-narrow": "R$"
  },
  "BRN": {
    "displayName": "Brasilianischer Cruzado Novo (1989-1990)",
    "displayName-count-one": "Brasilianischer Cruzado Novo (1989-
1990) ",
    "displayName-count-other": "Brasilianische Cruzado Novo (1989-
1990) ",
    "symbol": "BRN"
  },
  "BRR": {
    "displayName": "Brasilianischer Cruzeiro (1993-1994)",
    "displayName-count-one": "Brasilianischer Cruzeiro (1993-1994)",
    "displayName-count-other": "Brasilianische Cruzeiro (1993-
1994) ",
    "symbol": "BRR"
  },
  "BRZ": {
    "displayName": "Brasilianischer Cruzeiro (1942-1967)",
    "displayName-count-one": "Brasilianischer Cruzeiro (1942-1967)",
    "displayName-count-other": "Brasilianischer Cruzeiro (1942-
1967) ",
    "symbol": "BRZ"
  },
  "BSD": {
    "displayName": "Bahamas-Dollar",
    "displayName-count-one": "Bahamas-Dollar",
    "displayName-count-other": "Bahamas-Dollar",
    "symbol": "BSD",
    "symbol-alt-narrow": "$"
  },
  "BTN": {
    "displayName": "Bhutan-Ngultrum",

```



```

        "displayName-count-one": "Bhutan-Ngultrum",
        "displayName-count-other": "Bhutan-Ngultrum",
        "symbol": "BTN"
    },
    "BUK": {
        "displayName": "Birmanischer Kyat",
        "displayName-count-one": "Birmanischer Kyat",
        "displayName-count-other": "Birmanische Kyat",
        "symbol": "BUK"
    },
    "BWP": {
        "displayName": "Botswanischer Pula",
        "displayName-count-one": "Botswanischer Pula",
        "displayName-count-other": "Botswanische Pula",
        "symbol": "BWP",
        "symbol-alt-narrow": "P"
    },
    "BYB": {
        "displayName": "Belarus-Rubel (1994-1999)",
        "displayName-count-one": "Belarus-Rubel (1994-1999)",
        "displayName-count-other": "Belarus-Rubel (1994-1999)",
        "symbol": "BYB"
    },
    "BYN": {
        "displayName": "Weißrussischer Rubel",
        "displayName-count-one": "Weißrussischer Rubel",
        "displayName-count-other": "Weißrussische Rubel",
        "symbol": "BYN",
        "symbol-alt-narrow": "p."
    },
    "BYR": {
        "displayName": "Weißrussischer Rubel (2000-2016)",
        "displayName-count-one": "Weißrussischer Rubel (2000-2016)",
        "displayName-count-other": "Weißrussische Rubel (2000-2016)",
        "symbol": "BYR"
    },
    "BZD": {
        "displayName": "Belize-Dollar",
        "displayName-count-one": "Belize-Dollar",
        "displayName-count-other": "Belize-Dollar",
        "symbol": "BZD",
        "symbol-alt-narrow": "$"
    },
    "CAD": {
        "displayName": "Kanadischer Dollar",
        "displayName-count-one": "Kanadischer Dollar",
        "displayName-count-other": "Kanadische Dollar",
        "symbol": "CA$",
        "symbol-alt-narrow": "$"
    },
    "CDF": {
        "displayName": "Kongo-Franc",
        "displayName-count-one": "Kongo-Franc",
        "displayName-count-other": "Kongo-Francs",
        "symbol": "CDF"
    },
    "CHE": {

```

```

        "displayName": "WIR-Euro",
        "displayName-count-one": "WIR-Euro",
        "displayName-count-other": "WIR-Euro",
        "symbol": "CHE"
    },
    "CHF": {
        "displayName": "Schweizer Franken",
        "displayName-count-one": "Schweizer Franken",
        "displayName-count-other": "Schweizer Franken",
        "symbol": "CHF"
    },
    "CHW": {
        "displayName": "WIR Franken",
        "displayName-count-one": "WIR Franken",
        "displayName-count-other": "WIR Franken",
        "symbol": "CHW"
    },
    "CLE": {
        "displayName": "Chilenischer Escudo",
        "displayName-count-one": "Chilenischer Escudo",
        "displayName-count-other": "Chilenische Escudo",
        "symbol": "CLE"
    },
    "CLF": {
        "displayName": "Chilenische Unidades de Fomento",
        "displayName-count-one": "Chilenische Unidades de Fomento",
        "displayName-count-other": "Chilenische Unidades de Fomento",
        "symbol": "CLF"
    },
    "CLP": {
        "displayName": "Chilenischer Peso",
        "displayName-count-one": "Chilenischer Peso",
        "displayName-count-other": "Chilenische Pesos",
        "symbol": "CLP",
        "symbol-alt-narrow": "$"
    },
    "CNX": {
        "displayName": "Dollar der Chinesischen Volksbank",
        "displayName-count-one": "Dollar der Chinesischen Volksbank",
        "displayName-count-other": "Dollar der Chinesischen Volksbank",
        "symbol": "CNX"
    },
    "CNY": {
        "displayName": "Renminbi Yuan",
        "displayName-count-one": "Chinesischer Yuan",
        "displayName-count-other": "Renminbi Yuan",
        "symbol": "CN¥",
        "symbol-alt-narrow": "¥"
    },
    "COP": {
        "displayName": "Kolumbianischer Peso",
        "displayName-count-one": "Kolumbianischer Peso",
        "displayName-count-other": "Kolumbianische Pesos",
        "symbol": "COP",
        "symbol-alt-narrow": "$"
    },
    "COU": {

```

```

        "displayName": "Kolumbianische Unidades de valor real",
        "displayName-count-one": "Kolumbianische Unidad de valor real",
        "displayName-count-other": "Kolumbianische Unidades de valor
real",
        "symbol": "COU"
    },
    "CRC": {
        "displayName": "Costa-Rica-Colón",
        "displayName-count-one": "Costa-Rica-Colón",
        "displayName-count-other": "Costa-Rica-Colón",
        "symbol": "CRC",
        "symbol-alt-narrow": "₡"
    },
    "CSD": {
        "displayName": "Serbischer Dinar (2002-2006)",
        "displayName-count-one": "Serbischer Dinar (2002-2006)",
        "displayName-count-other": "Serbische Dinar (2002-2006)",
        "symbol": "CSD"
    },
    "CSK": {
        "displayName": "Tschechoslowakische Krone",
        "displayName-count-one": "Tschechoslowakische Kronen",
        "displayName-count-other": "Tschechoslowakische Kronen",
        "symbol": "CSK"
    },
    "CUC": {
        "displayName": "Kubanischer Peso (konvertibel)",
        "displayName-count-one": "Kubanischer Peso (konvertibel)",
        "displayName-count-other": "Kubanische Pesos (konvertibel)",
        "symbol": "CUC",
        "symbol-alt-narrow": "Cub$"
    },
    "CUP": {
        "displayName": "Kubanischer Peso",
        "displayName-count-one": "Kubanischer Peso",
        "displayName-count-other": "Kubanische Pesos",
        "symbol": "CUP",
        "symbol-alt-narrow": "$"
    },
    "CVE": {
        "displayName": "Cabo-Verde-Escudo",
        "displayName-count-one": "Cabo-Verde-Escudo",
        "displayName-count-other": "Cabo-Verde-Escudos",
        "symbol": "CVE"
    },
    "CYP": {
        "displayName": "Zypern-Pfund",
        "displayName-count-one": "Zypern Pfund",
        "displayName-count-other": "Zypern Pfund",
        "symbol": "CYP"
    },
    "CZK": {
        "displayName": "Tschechische Krone",
        "displayName-count-one": "Tschechische Krone",
        "displayName-count-other": "Tschechische Kronen",
        "symbol": "CZK",
        "symbol-alt-narrow": "Kč"
    }

```

```

    },
    "DDM": {
      "displayName": "Mark der DDR",
      "displayName-count-one": "Mark der DDR",
      "displayName-count-other": "Mark der DDR",
      "symbol": "DDM"
    },
    "DEM": {
      "displayName": "Deutsche Mark",
      "displayName-count-one": "Deutsche Mark",
      "displayName-count-other": "Deutsche Mark",
      "symbol": "DM"
    },
    "DJF": {
      "displayName": "Dschibuti-Franc",
      "displayName-count-one": "Dschibuti-Franc",
      "displayName-count-other": "Dschibuti-Franc",
      "symbol": "DJF"
    },
    "DKK": {
      "displayName": "Dänische Krone",
      "displayName-count-one": "Dänische Krone",
      "displayName-count-other": "Dänische Kronen",
      "symbol": "DKK",
      "symbol-alt-narrow": "kr"
    },
    "DOP": {
      "displayName": "Dominikanischer Peso",
      "displayName-count-one": "Dominikanischer Peso",
      "displayName-count-other": "Dominikanische Pesos",
      "symbol": "DOP",
      "symbol-alt-narrow": "$"
    },
    "DZD": {
      "displayName": "Algerischer Dinar",
      "displayName-count-one": "Algerischer Dinar",
      "displayName-count-other": "Algerische Dinar",
      "symbol": "DZD"
    },
    "ECS": {
      "displayName": "Ecuadorianischer Sucre",
      "displayName-count-one": "Ecuadorianischer Sucre",
      "displayName-count-other": "Ecuadorianische Sucre",
      "symbol": "ECS"
    },
    "ECV": {
      "displayName": "Verrechnungseinheit für Ecuador",
      "displayName-count-one": "Verrechnungseinheiten für Ecuador",
      "displayName-count-other": "Verrechnungseinheiten für Ecuador",
      "symbol": "ECV"
    },
    "EEK": {
      "displayName": "Estnische Krone",
      "displayName-count-one": "Estnische Krone",
      "displayName-count-other": "Estnische Kronen",
      "symbol": "EEK"
    },
  },

```

```

"EGP": {
  "displayName": "Ägyptisches Pfund",
  "displayName-count-one": "Ägyptisches Pfund",
  "displayName-count-other": "Ägyptische Pfund",
  "symbol": "EGP",
  "symbol-alt-narrow": "E£"
},
"ERN": {
  "displayName": "Eritreischer Nakfa",
  "displayName-count-one": "Eritreischer Nakfa",
  "displayName-count-other": "Eritreische Nakfa",
  "symbol": "ERN"
},
"ESA": {
  "displayName": "Spanische Peseta (A-Konten)",
  "displayName-count-one": "Spanische Peseta (A-Konten)",
  "displayName-count-other": "Spanische Peseten (A-Konten)",
  "symbol": "ESA"
},
"ESB": {
  "displayName": "Spanische Peseta (konvertibel)",
  "displayName-count-one": "Spanische Peseta (konvertibel)",
  "displayName-count-other": "Spanische Peseten (konvertibel)",
  "symbol": "ESB"
},
"ESP": {
  "displayName": "Spanische Peseta",
  "displayName-count-one": "Spanische Peseta",
  "displayName-count-other": "Spanische Peseten",
  "symbol": "ESP",
  "symbol-alt-narrow": "₧"
},
"ETB": {
  "displayName": "Äthiopischer Birr",
  "displayName-count-one": "Äthiopischer Birr",
  "displayName-count-other": "Äthiopische Birr",
  "symbol": "ETB"
},
"EUR": {
  "displayName": "Euro",
  "displayName-count-one": "Euro",
  "displayName-count-other": "Euro",
  "symbol": "€",
  "symbol-alt-narrow": "€"
},
"FIM": {
  "displayName": "Finnische Mark",
  "displayName-count-one": "Finnische Mark",
  "displayName-count-other": "Finnische Mark",
  "symbol": "FIM"
},
"FJD": {
  "displayName": "Fidschi-Dollar",
  "displayName-count-one": "Fidschi-Dollar",
  "displayName-count-other": "Fidschi-Dollar",
  "symbol": "FJD",
  "symbol-alt-narrow": "$"
}

```

```

    },
    "FKP": {
      "displayName": "Falkland-Pfund",
      "displayName-count-one": "Falkland-Pfund",
      "displayName-count-other": "Falkland-Pfund",
      "symbol": "FKP",
      "symbol-alt-narrow": "Fl£"
    },
    "FRF": {
      "displayName": "Französischer Franc",
      "displayName-count-one": "Französischer Franc",
      "displayName-count-other": "Französische Franc",
      "symbol": "FRF"
    },
    "GBP": {
      "displayName": "Britisches Pfund",
      "displayName-count-one": "Britisches Pfund",
      "displayName-count-other": "Britische Pfund",
      "symbol": "£",
      "symbol-alt-narrow": "£"
    },
    "GEK": {
      "displayName": "Georgischer Kupon Larit",
      "displayName-count-one": "Georgischer Kupon Larit",
      "displayName-count-other": "Georgische Kupon Larit",
      "symbol": "GEK"
    },
    "GEL": {
      "displayName": "Georgischer Lari",
      "displayName-count-one": "Georgischer Lari",
      "displayName-count-other": "Georgische Lari",
      "symbol": "GEL",
      "symbol-alt-narrow": "ლ",
      "symbol-alt-variant": "ლ"
    },
    "GHC": {
      "displayName": "Ghanaischer Cedi (1979-2007)",
      "displayName-count-one": "Ghanaischer Cedi (1979-2007)",
      "displayName-count-other": "Ghanaische Cedi (1979-2007)",
      "symbol": "GHC"
    },
    "GHS": {
      "displayName": "Ghanaischer Cedi",
      "displayName-count-one": "Ghanaischer Cedi",
      "displayName-count-other": "Ghanaische Cedi",
      "symbol": "GHS"
    },
    "GIP": {
      "displayName": "Gibraltar-Pfund",
      "displayName-count-one": "Gibraltar-Pfund",
      "displayName-count-other": "Gibraltar Pfund",
      "symbol": "GIP",
      "symbol-alt-narrow": "£"
    },
    "GMD": {
      "displayName": "Gambia-Dalasi",
      "displayName-count-one": "Gambia-Dalasi",

```

```

        "displayName-count-other": "Gambia-Dalasi",
        "symbol": "GMD"
    },
    "GNF": {
        "displayName": "Guinea-Franc",
        "displayName-count-one": "Guinea-Franc",
        "displayName-count-other": "Guinea-Franc",
        "symbol": "GNF",
        "symbol-alt-narrow": "F.G."
    },
    "GNS": {
        "displayName": "Guineischer Syli",
        "displayName-count-one": "Guineischer Syli",
        "displayName-count-other": "Guineische Syli",
        "symbol": "GNS"
    },
    "GQE": {
        "displayName": "Äquatorialguinea-Ekwele",
        "displayName-count-one": "Äquatorialguinea-Ekwele",
        "displayName-count-other": "Äquatorialguinea-Ekwele",
        "symbol": "GQE"
    },
    "GRD": {
        "displayName": "Griechische Drachme",
        "displayName-count-one": "Griechische Drachme",
        "displayName-count-other": "Griechische Drachmen",
        "symbol": "GRD"
    },
    "GTQ": {
        "displayName": "Guatemaltekinscher Quetzal",
        "displayName-count-one": "Guatemaltekinscher Quetzal",
        "displayName-count-other": "Guatemaltekinsche Quetzales",
        "symbol": "GTQ",
        "symbol-alt-narrow": "Q"
    },
    "GWE": {
        "displayName": "Portugiesisch Guinea Escudo",
        "displayName-count-one": "Portugiesisch Guinea Escudo",
        "displayName-count-other": "Portugiesisch Guinea Escudo",
        "symbol": "GWE"
    },
    "GWP": {
        "displayName": "Guinea-Bissau Peso",
        "displayName-count-one": "Guinea-Bissau Peso",
        "displayName-count-other": "Guinea-Bissau Pesos",
        "symbol": "GWP"
    },
    "GYD": {
        "displayName": "Guyana-Dollar",
        "displayName-count-one": "Guyana-Dollar",
        "displayName-count-other": "Guyana-Dollar",
        "symbol": "GYD",
        "symbol-alt-narrow": "$"
    },
    "HKD": {
        "displayName": "Hongkong-Dollar",
        "displayName-count-one": "Hongkong-Dollar",

```

```

        "displayName-count-other": "Hongkong-Dollar",
        "symbol": "HK$",
        "symbol-alt-narrow": "$"
    },
    "HNL": {
        "displayName": "Honduras-Lempira",
        "displayName-count-one": "Honduras-Lempira",
        "displayName-count-other": "Honduras-Lempira",
        "symbol": "HNL",
        "symbol-alt-narrow": "L"
    },
    "HRD": {
        "displayName": "Kroatischer Dinar",
        "displayName-count-one": "Kroatischer Dinar",
        "displayName-count-other": "Kroatische Dinar",
        "symbol": "HRD"
    },
    "HRK": {
        "displayName": "Kroatischer Kuna",
        "displayName-count-one": "Kroatischer Kuna",
        "displayName-count-other": "Kroatische Kuna",
        "symbol": "HRK",
        "symbol-alt-narrow": "kn"
    },
    "HTG": {
        "displayName": "Haitianische Gourde",
        "displayName-count-one": "Haitianische Gourde",
        "displayName-count-other": "Haitianische Gourdes",
        "symbol": "HTG"
    },
    "HUF": {
        "displayName": "Ungarischer Forint",
        "displayName-count-one": "Ungarischer Forint",
        "displayName-count-other": "Ungarische Forint",
        "symbol": "HUF",
        "symbol-alt-narrow": "Ft"
    },
    "IDR": {
        "displayName": "Indonesische Rupiah",
        "displayName-count-one": "Indonesische Rupiah",
        "displayName-count-other": "Indonesische Rupiah",
        "symbol": "IDR",
        "symbol-alt-narrow": "Rp"
    },
    "IEP": {
        "displayName": "Irishes Pfund",
        "displayName-count-one": "Irishes Pfund",
        "displayName-count-other": "Irische Pfund",
        "symbol": "IEP"
    },
    "ILP": {
        "displayName": "Israelisches Pfund",
        "displayName-count-one": "Israelisches Pfund",
        "displayName-count-other": "Israelische Pfund",
        "symbol": "ILP"
    },
    "ILR": {

```



```

        "displayName": "Israelischer Schekel (1980-1985)",
        "displayName-count-one": "Israelischer Schekel (1980-1985)",
        "displayName-count-other": "Israelische Schekel (1980-1985)"
    },
    "ILS": {
        "displayName": "Israelischer Neuer Schekel",
        "displayName-count-one": "Israelischer Neuer Schekel",
        "displayName-count-other": "Israelische Neue Schekel",
        "symbol": "₪",
        "symbol-alt-narrow": "₪"
    },
    "INR": {
        "displayName": "Indische Rupie",
        "displayName-count-one": "Indische Rupie",
        "displayName-count-other": "Indische Rupien",
        "symbol": "₹",
        "symbol-alt-narrow": "₹"
    },
    "IQD": {
        "displayName": "Irakischer Dinar",
        "displayName-count-one": "Irakischer Dinar",
        "displayName-count-other": "Irakische Dinar",
        "symbol": "IQD"
    },
    "IRR": {
        "displayName": "Iranischer Rial",
        "displayName-count-one": "Iranischer Rial",
        "displayName-count-other": "Iranische Rial",
        "symbol": "IRR"
    },
    "ISJ": {
        "displayName": "Isländische Krone (1918-1981)",
        "displayName-count-one": "Isländische Krone (1918-1981)",
        "displayName-count-other": "Isländische Kronen (1918-1981)"
    },
    "ISK": {
        "displayName": "Isländische Krone",
        "displayName-count-one": "Isländische Krone",
        "displayName-count-other": "Isländische Kronen",
        "symbol": "ISK",
        "symbol-alt-narrow": "kr"
    },
    "ITL": {
        "displayName": "Italienische Lira",
        "displayName-count-one": "Italienische Lira",
        "displayName-count-other": "Italienische Lire",
        "symbol": "ITL"
    },
    "JMD": {
        "displayName": "Jamaika-Dollar",
        "displayName-count-one": "Jamaika-Dollar",
        "displayName-count-other": "Jamaika-Dollar",
        "symbol": "JMD",
        "symbol-alt-narrow": "$"
    },
    "JOD": {
        "displayName": "Jordanischer Dinar",

```

```

        "displayName-count-one": "Jordanischer Dinar",
        "displayName-count-other": "Jordanische Dinar",
        "symbol": "JOD"
    },
    "JPY": {
        "displayName": "Japanischer Yen",
        "displayName-count-one": "Japanischer Yen",
        "displayName-count-other": "Japanische Yen",
        "symbol": "¥",
        "symbol-alt-narrow": "¥"
    },
    "KES": {
        "displayName": "Kenia-Schilling",
        "displayName-count-one": "Kenia-Schilling",
        "displayName-count-other": "Kenia-Schilling",
        "symbol": "KES"
    },
    "KGS": {
        "displayName": "Kirgisischer Som",
        "displayName-count-one": "Kirgisischer Som",
        "displayName-count-other": "Kirgisische Som",
        "symbol": "KGS"
    },
    "KHR": {
        "displayName": "Kambodschanischer Riel",
        "displayName-count-one": "Kambodschanischer Riel",
        "displayName-count-other": "Kambodschanische Riel",
        "symbol": "KHR",
        "symbol-alt-narrow": "៛"
    },
    "KMF": {
        "displayName": "Komoren-Franc",
        "displayName-count-one": "Komoren-Franc",
        "displayName-count-other": "Komoren-Francs",
        "symbol": "KMF",
        "symbol-alt-narrow": "FC"
    },
    "KPW": {
        "displayName": "Nordkoreanischer Won",
        "displayName-count-one": "Nordkoreanischer Won",
        "displayName-count-other": "Nordkoreanische Won",
        "symbol": "KPW",
        "symbol-alt-narrow": "₩"
    },
    "KRH": {
        "displayName": "Südkoreanischer Hwan (1953-1962)",
        "displayName-count-one": "Südkoreanischer Hwan (1953-1962)",
        "displayName-count-other": "Südkoreanischer Hwan (1953-1962)",
        "symbol": "KRH"
    },
    "KRO": {
        "displayName": "Südkoreanischer Won (1945-1953)",
        "displayName-count-one": "Südkoreanischer Won (1945-1953)",
        "displayName-count-other": "Südkoreanischer Won (1945-1953)",
        "symbol": "KRO"
    },
    },

```

```

"KRW": {
  "displayName": "Südkoreanischer Won",
  "displayName-count-one": "Südkoreanischer Won",
  "displayName-count-other": "Südkoreanische Won",
  "symbol": "₩",
  "symbol-alt-narrow": "₩"
},
"KWD": {
  "displayName": "Kuwait-Dinar",
  "displayName-count-one": "Kuwait-Dinar",
  "displayName-count-other": "Kuwait-Dinar",
  "symbol": "KWD"
},
"KYD": {
  "displayName": "Kaiman-Dollar",
  "displayName-count-one": "Kaiman-Dollar",
  "displayName-count-other": "Kaiman-Dollar",
  "symbol": "KYD",
  "symbol-alt-narrow": "$"
},
"KZT": {
  "displayName": "Kasachischer Tenge",
  "displayName-count-one": "Kasachischer Tenge",
  "displayName-count-other": "Kasachische Tenge",
  "symbol": "KZT",
  "symbol-alt-narrow": "T"
},
"LAK": {
  "displayName": "Laotischer Kip",
  "displayName-count-one": "Laotischer Kip",
  "displayName-count-other": "Laotische Kip",
  "symbol": "LAK",
  "symbol-alt-narrow": "₭"
},
"LBP": {
  "displayName": "Libanesisches Pfund",
  "displayName-count-one": "Libanesisches Pfund",
  "displayName-count-other": "Libanesische Pfund",
  "symbol": "LBP",
  "symbol-alt-narrow": "L£"
},
"LKR": {
  "displayName": "Sri-Lanka-Rupie",
  "displayName-count-one": "Sri-Lanka-Rupie",
  "displayName-count-other": "Sri-Lanka-Rupien",
  "symbol": "LKR",
  "symbol-alt-narrow": "Rs"
},
"LRD": {
  "displayName": "Liberianischer Dollar",
  "displayName-count-one": "Liberianischer Dollar",
  "displayName-count-other": "Liberianische Dollar",
  "symbol": "LRD",
  "symbol-alt-narrow": "$"
},
"LSL": {
  "displayName": "Loti",

```

```

        "displayName-count-one": "Loti",
        "displayName-count-other": "Loti",
        "symbol": "LSL"
    },
    "LTL": {
        "displayName": "Litauischer Litas",
        "displayName-count-one": "Litauischer Litas",
        "displayName-count-other": "Litauische Litas",
        "symbol": "LTL",
        "symbol-alt-narrow": "Lt"
    },
    "LTT": {
        "displayName": "Litauischer Talonas",
        "displayName-count-one": "Litauische Talonas",
        "displayName-count-other": "Litauische Talonas",
        "symbol": "LTT"
    },
    "LUC": {
        "displayName": "Luxemburgischer Franc (konvertibel)",
        "displayName-count-one": "Luxemburgische Franc (konvertibel)",
        "displayName-count-other": "Luxemburgische Franc (konvertibel)",
        "symbol": "LUC"
    },
    "LUF": {
        "displayName": "Luxemburgischer Franc",
        "displayName-count-one": "Luxemburgische Franc",
        "displayName-count-other": "Luxemburgische Franc",
        "symbol": "LUF"
    },
    "LUL": {
        "displayName": "Luxemburgischer Finanz-Franc",
        "displayName-count-one": "Luxemburgische Finanz-Franc",
        "displayName-count-other": "Luxemburgische Finanz-Franc",
        "symbol": "LUL"
    },
    "LVL": {
        "displayName": "Lettischer Lats",
        "displayName-count-one": "Lettischer Lats",
        "displayName-count-other": "Lettische Lats",
        "symbol": "LVL",
        "symbol-alt-narrow": "Ls"
    },
    "LVR": {
        "displayName": "Lettischer Rubel",
        "displayName-count-one": "Lettische Rubel",
        "displayName-count-other": "Lettische Rubel",
        "symbol": "LVR"
    },
    "LYD": {
        "displayName": "Libyscher Dinar",
        "displayName-count-one": "Libyscher Dinar",
        "displayName-count-other": "Libysche Dinar",
        "symbol": "LYD"
    },
    "MAD": {
        "displayName": "Marokkanischer Dirham",
        "displayName-count-one": "Marokkanischer Dirham",

```

```

    "displayName-count-other": "Marokkanische Dirham",
    "symbol": "MAD"
  },
  "MAF": {
    "displayName": "Marokkanischer Franc",
    "displayName-count-one": "Marokkanische Franc",
    "displayName-count-other": "Marokkanische Franc",
    "symbol": "MAF"
  },
  "MCF": {
    "displayName": "Monegassischer Franc",
    "displayName-count-one": "Monegassischer Franc",
    "displayName-count-other": "Monegassische Franc",
    "symbol": "MCF"
  },
  "MDC": {
    "displayName": "Moldau-Cupon",
    "displayName-count-one": "Moldau-Cupon",
    "displayName-count-other": "Moldau-Cupon",
    "symbol": "MDC"
  },
  "MDL": {
    "displayName": "Moldau-Leu",
    "displayName-count-one": "Moldau-Leu",
    "displayName-count-other": "Moldau-Leu",
    "symbol": "MDL"
  },
  "MGA": {
    "displayName": "Madagaskar-Ariary",
    "displayName-count-one": "Madagaskar-Ariary",
    "displayName-count-other": "Madagaskar-Ariary",
    "symbol": "MGA",
    "symbol-alt-narrow": "Ar"
  },
  "MGF": {
    "displayName": "Madagaskar-Franc",
    "displayName-count-one": "Madagaskar-Franc",
    "displayName-count-other": "Madagaskar-Franc",
    "symbol": "MGF"
  },
  "MKD": {
    "displayName": "Mazedonischer Denar",
    "displayName-count-one": "Mazedonischer Denar",
    "displayName-count-other": "Mazedonische Denari",
    "symbol": "MKD"
  },
  "MKN": {
    "displayName": "Mazedonischer Denar (1992-1993)",
    "displayName-count-one": "Mazedonischer Denar (1992-1993)",
    "displayName-count-other": "Mazedonische Denar (1992-1993)",
    "symbol": "MKN"
  },
  "MLF": {
    "displayName": "Malischer Franc",
    "displayName-count-one": "Malische Franc",
    "displayName-count-other": "Malische Franc",
    "symbol": "MLF"
  }

```

```

    },
    "MMK": {
      "displayName": "Myanmarischer Kyat",
      "displayName-count-one": "Myanmarischer Kyat",
      "displayName-count-other": "Myanmarische Kyat",
      "symbol": "MMK",
      "symbol-alt-narrow": "K"
    },
    "MNT": {
      "displayName": "Mongolischer Tögrög",
      "displayName-count-one": "Mongolischer Tögrög",
      "displayName-count-other": "Mongolische Tögrög",
      "symbol": "MNT",
      "symbol-alt-narrow": "₮"
    },
    "MOP": {
      "displayName": "Macao-Pataca",
      "displayName-count-one": "Macao-Pataca",
      "displayName-count-other": "Macao-Pataca",
      "symbol": "MOP"
    },
    "MRO": {
      "displayName": "Mauretanischer Ouguiya",
      "displayName-count-one": "Mauretanischer Ouguiya",
      "displayName-count-other": "Mauretanische Ouguiya",
      "symbol": "MRO"
    },
    "MTL": {
      "displayName": "Maltesische Lira",
      "displayName-count-one": "Maltesische Lira",
      "displayName-count-other": "Maltesische Lira",
      "symbol": "MTL"
    },
    "MTP": {
      "displayName": "Maltesisches Pfund",
      "displayName-count-one": "Maltesische Pfund",
      "displayName-count-other": "Maltesische Pfund",
      "symbol": "MTP"
    },
    "MUR": {
      "displayName": "Mauritius-Rupie",
      "displayName-count-one": "Mauritius-Rupie",
      "displayName-count-other": "Mauritius-Rupien",
      "symbol": "MUR",
      "symbol-alt-narrow": "Rs"
    },
    "MVP": {
      "displayName": "Malediven-Rupie (alt)",
      "displayName-count-one": "Malediven-Rupie (alt)",
      "displayName-count-other": "Malediven-Rupien (alt)"
    },
    "MVR": {
      "displayName": "Malediven-Rufiyaa",
      "displayName-count-one": "Malediven-Rufiyaa",
      "displayName-count-other": "Malediven-Rupien",
      "symbol": "MVR"
    },
  },

```

```

    "MWK": {
      "displayName": "Malawi-Kwacha",
      "displayName-count-one": "Malawi-Kwacha",
      "displayName-count-other": "Malawi-Kwacha",
      "symbol": "MWK"
    },
    "MXN": {
      "displayName": "Mexikanischer Peso",
      "displayName-count-one": "Mexikanischer Peso",
      "displayName-count-other": "Mexikanische Pesos",
      "symbol": "MX$",
      "symbol-alt-narrow": "$"
    },
    "MXP": {
      "displayName": "Mexikanischer Silber-Peso (1861-1992)",
      "displayName-count-one": "Mexikanische Silber-Peso (1861-1992)",
      "displayName-count-other": "Mexikanische Silber-Pesos (1861-
1992)",
      "symbol": "MXP"
    },
    "MXV": {
      "displayName": "Mexicanischer Unidad de Inversion (UDI)",
      "displayName-count-one": "Mexicanischer Unidad de Inversion
(UDI)",
      "displayName-count-other": "Mexikanische Unidad de Inversion
(UDI)",
      "symbol": "MXV"
    },
    "MYR": {
      "displayName": "Malaysischer Ringgit",
      "displayName-count-one": "Malaysischer Ringgit",
      "displayName-count-other": "Malaysische Ringgit",
      "symbol": "MYR",
      "symbol-alt-narrow": "RM"
    },
    "MZE": {
      "displayName": "Mosambikanischer Escudo",
      "displayName-count-one": "Mozambikanische Escudo",
      "displayName-count-other": "Mozambikanische Escudo",
      "symbol": "MZE"
    },
    "MZM": {
      "displayName": "Mosambikanischer Metical (1980-2006)",
      "displayName-count-one": "Mosambikanischer Metical (1980-2006)",
      "displayName-count-other": "Mosambikanische Meticais (1980-
2006)",
      "symbol": "MZM"
    },
    "MZN": {
      "displayName": "Mosambikanischer Metical",
      "displayName-count-one": "Mosambikanischer Metical",
      "displayName-count-other": "Mosambikanische Meticais",
      "symbol": "MZN"
    },
    "NAD": {
      "displayName": "Namibia-Dollar",
      "displayName-count-one": "Namibia-Dollar",

```

```

        "displayName-count-other": "Namibia-Dollar",
        "symbol": "NAD",
        "symbol-alt-narrow": "$"
    },
    "NGN": {
        "displayName": "Nigerianischer Naira",
        "displayName-count-one": "Nigerianischer Naira",
        "displayName-count-other": "Nigerianische Naira",
        "symbol": "NGN",
        "symbol-alt-narrow": "₦"
    },
    "NIC": {
        "displayName": "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-one": "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-other": "Nicaraguanische Córdoba (1988-
1991)",
        "symbol": "NIC"
    },
    "NIO": {
        "displayName": "Nicaragua-Córdoba",
        "displayName-count-one": "Nicaragua-Córdoba",
        "displayName-count-other": "Nicaragua-Córdobas",
        "symbol": "NIO",
        "symbol-alt-narrow": "C$"
    },
    "NLG": {
        "displayName": "Niederländischer Gulden",
        "displayName-count-one": "Niederländischer Gulden",
        "displayName-count-other": "Niederländische Gulden",
        "symbol": "NLG"
    },
    "NOK": {
        "displayName": "Norwegische Krone",
        "displayName-count-one": "Norwegische Krone",
        "displayName-count-other": "Norwegische Kronen",
        "symbol": "NOK",
        "symbol-alt-narrow": "kr"
    },
    "NPR": {
        "displayName": "Nepalesische Rupie",
        "displayName-count-one": "Nepalesische Rupie",
        "displayName-count-other": "Nepalesische Rupien",
        "symbol": "NPR",
        "symbol-alt-narrow": "Rs"
    },
    "NZD": {
        "displayName": "Neuseeland-Dollar",
        "displayName-count-one": "Neuseeland-Dollar",
        "displayName-count-other": "Neuseeland-Dollar",
        "symbol": "NZ$",
        "symbol-alt-narrow": "$"
    },
    "OMR": {
        "displayName": "Omanischer Rial",
        "displayName-count-one": "Omanischer Rial",
        "displayName-count-other": "Omanische Rials",
        "symbol": "OMR"
    }

```



```

    },
    "PAB": {
      "displayName": "Panamaischer Balboa",
      "displayName-count-one": "Panamaischer Balboa",
      "displayName-count-other": "Panamaische Balboas",
      "symbol": "PAB"
    },
    "PEI": {
      "displayName": "Peruanischer Inti",
      "displayName-count-one": "Peruanische Inti",
      "displayName-count-other": "Peruanische Inti",
      "symbol": "PEI"
    },
    "PEN": {
      "displayName": "Peruanischer Sol",
      "displayName-count-one": "Peruanischer Sol",
      "displayName-count-other": "Peruanische Sol",
      "symbol": "PEN"
    },
    "PES": {
      "displayName": "Peruanischer Sol (1863-1965)",
      "displayName-count-one": "Peruanischer Sol (1863-1965)",
      "displayName-count-other": "Peruanische Sol (1863-1965)",
      "symbol": "PES"
    },
    "PGK": {
      "displayName": "Papua-Neuguineischer Kina",
      "displayName-count-one": "Papua-Neuguineischer Kina",
      "displayName-count-other": "Papua-Neuguineische Kina",
      "symbol": "PGK"
    },
    "PHP": {
      "displayName": "Philippinischer Peso",
      "displayName-count-one": "Philippinischer Peso",
      "displayName-count-other": "Philippinische Pesos",
      "symbol": "PHP",
      "symbol-alt-narrow": "₱"
    },
    "PKR": {
      "displayName": "Pakistanische Rupie",
      "displayName-count-one": "Pakistanische Rupie",
      "displayName-count-other": "Pakistanische Rupien",
      "symbol": "PKR",
      "symbol-alt-narrow": "Rs"
    },
    "PLN": {
      "displayName": "Polnischer Złoty",
      "displayName-count-one": "Polnischer Złoty",
      "displayName-count-other": "Polnische Złoty",
      "symbol": "PLN",
      "symbol-alt-narrow": "zł"
    },
    "PLZ": {
      "displayName": "Polnischer Zloty (1950-1995)",
      "displayName-count-one": "Polnischer Zloty (1950-1995)",
      "displayName-count-other": "Polnische Zloty (1950-1995)",
      "symbol": "PLZ"
    }
  }

```

```

    },
    "PTE": {
      "displayName": "Portugiesischer Escudo",
      "displayName-count-one": "Portugiesische Escudo",
      "displayName-count-other": "Portugiesische Escudo",
      "symbol": "PTE"
    },
    "PYG": {
      "displayName": "Paraguayischer Guaraní",
      "displayName-count-one": "Paraguayischer Guaraní",
      "displayName-count-other": "Paraguayische Guaraníes",
      "symbol": "PYG",
      "symbol-alt-narrow": "₲"
    },
    "QAR": {
      "displayName": "Katar-Riyal",
      "displayName-count-one": "Katar-Riyal",
      "displayName-count-other": "Katar-Riyal",
      "symbol": "QAR"
    },
    "RHD": {
      "displayName": "Rhodesischer Dollar",
      "displayName-count-one": "Rhodesische Dollar",
      "displayName-count-other": "Rhodesische Dollar",
      "symbol": "RHD"
    },
    "ROL": {
      "displayName": "Rumänischer Leu (1952-2006)",
      "displayName-count-one": "Rumänischer Leu (1952-2006)",
      "displayName-count-other": "Rumänische Leu (1952-2006)",
      "symbol": "ROL"
    },
    "RON": {
      "displayName": "Rumänischer Leu",
      "displayName-count-one": "Rumänischer Leu",
      "displayName-count-other": "Rumänische Leu",
      "symbol": "RON",
      "symbol-alt-narrow": "L"
    },
    "RSD": {
      "displayName": "Serbischer Dinar",
      "displayName-count-one": "Serbischer Dinar",
      "displayName-count-other": "Serbische Dinaren",
      "symbol": "RSD"
    },
    "RUB": {
      "displayName": "Russischer Rubel",
      "displayName-count-one": "Russischer Rubel",
      "displayName-count-other": "Russische Rubel",
      "symbol": "RUB",
      "symbol-alt-narrow": "₽"
    },
    "RUR": {
      "displayName": "Russischer Rubel (1991-1998)",
      "displayName-count-one": "Russischer Rubel (1991-1998)",
      "displayName-count-other": "Russische Rubel (1991-1998)",
      "symbol": "RUR",

```

```

    "symbol-alt-narrow": "p."
  },
  "RWF": {
    "displayName": "Ruanda-Franc",
    "displayName-count-one": "Ruanda-Franc",
    "displayName-count-other": "Ruanda-Francs",
    "symbol": "RWF",
    "symbol-alt-narrow": "F.Rw"
  },
  "SAR": {
    "displayName": "Saudi-Rial",
    "displayName-count-one": "Saudi-Rial",
    "displayName-count-other": "Saudi-Rial",
    "symbol": "SAR"
  },
  "SBD": {
    "displayName": "Salomonen-Dollar",
    "displayName-count-one": "Salomonen-Dollar",
    "displayName-count-other": "Salomonen-Dollar",
    "symbol": "SBD",
    "symbol-alt-narrow": "$"
  },
  "SCR": {
    "displayName": "Seychellen-Rupie",
    "displayName-count-one": "Seychellen-Rupie",
    "displayName-count-other": "Seychellen-Rupien",
    "symbol": "SCR"
  },
  "SDD": {
    "displayName": "Sudanesischer Dinar (1992-2007)",
    "displayName-count-one": "Sudanesischer Dinar (1992-2007)",
    "displayName-count-other": "Sudanesische Dinar (1992-2007)",
    "symbol": "SDD"
  },
  "SDG": {
    "displayName": "Sudanesisches Pfund",
    "displayName-count-one": "Sudanesisches Pfund",
    "displayName-count-other": "Sudanesische Pfund",
    "symbol": "SDG"
  },
  "SDP": {
    "displayName": "Sudanesisches Pfund (1957-1998)",
    "displayName-count-one": "Sudanesisches Pfund (1957-1998)",
    "displayName-count-other": "Sudanesische Pfund (1957-1998)",
    "symbol": "SDP"
  },
  "SEK": {
    "displayName": "Schwedische Krone",
    "displayName-count-one": "Schwedische Krone",
    "displayName-count-other": "Schwedische Kronen",
    "symbol": "SEK",
    "symbol-alt-narrow": "kr"
  },
  "SGD": {
    "displayName": "Singapur-Dollar",
    "displayName-count-one": "Singapur-Dollar",
    "displayName-count-other": "Singapur-Dollar",

```

```

        "symbol": "SGD",
        "symbol-alt-narrow": "$"
    },
    "SHP": {
        "displayName": "St. Helena-Pfund",
        "displayName-count-one": "St. Helena-Pfund",
        "displayName-count-other": "St. Helena-Pfund",
        "symbol": "SHP",
        "symbol-alt-narrow": "£"
    },
    "SIT": {
        "displayName": "Slowenischer Tolar",
        "displayName-count-one": "Slowenischer Tolar",
        "displayName-count-other": "Slowenische Tolar",
        "symbol": "SIT"
    },
    "SKK": {
        "displayName": "Slowakische Krone",
        "displayName-count-one": "Slowakische Kronen",
        "displayName-count-other": "Slowakische Kronen",
        "symbol": "SKK"
    },
    "SLL": {
        "displayName": "Sierra-leonischer Leone",
        "displayName-count-one": "Sierra-leonischer Leone",
        "displayName-count-other": "Sierra-leonische Leones",
        "symbol": "SLL"
    },
    "SOS": {
        "displayName": "Somalia-Schilling",
        "displayName-count-one": "Somalia-Schilling",
        "displayName-count-other": "Somalia-Schilling",
        "symbol": "SOS"
    },
    "SRD": {
        "displayName": "Suriname-Dollar",
        "displayName-count-one": "Suriname-Dollar",
        "displayName-count-other": "Suriname-Dollar",
        "symbol": "SRD",
        "symbol-alt-narrow": "$"
    },
    "SRG": {
        "displayName": "Suriname Gulden",
        "displayName-count-one": "Suriname-Gulden",
        "displayName-count-other": "Suriname-Gulden",
        "symbol": "SRG"
    },
    "SSP": {
        "displayName": "Südsudanesisches Pfund",
        "displayName-count-one": "Südsudanesisches Pfund",
        "displayName-count-other": "Südsudanesische Pfund",
        "symbol": "SSP",
        "symbol-alt-narrow": "£"
    },
    "STD": {
        "displayName": "São-toméischer Dobra",
        "displayName-count-one": "São-toméischer Dobra",

```

```

    "displayName-count-other": "São-toméische Dobra",
    "symbol": "STD",
    "symbol-alt-narrow": "Db"
  },
  "SUR": {
    "displayName": "Sowjetischer Rubel",
    "displayName-count-one": "Sowjetische Rubel",
    "displayName-count-other": "Sowjetische Rubel",
    "symbol": "SUR"
  },
  "SVC": {
    "displayName": "El Salvador Colon",
    "displayName-count-one": "El Salvador-Colon",
    "displayName-count-other": "El Salvador-Colon",
    "symbol": "SVC"
  },
  "SYP": {
    "displayName": "Syrisches Pfund",
    "displayName-count-one": "Syrisches Pfund",
    "displayName-count-other": "Syrische Pfund",
    "symbol": "SYP",
    "symbol-alt-narrow": "SYP"
  },
  "SZL": {
    "displayName": "Swasiländischer Lilangeni",
    "displayName-count-one": "Swasiländischer Lilangeni",
    "displayName-count-other": "Swasiländische Emalangeni",
    "symbol": "SZL"
  },
  "THB": {
    "displayName": "Thailändischer Baht",
    "displayName-count-one": "Thailändischer Baht",
    "displayName-count-other": "Thailändische Baht",
    "symbol": "฿",
    "symbol-alt-narrow": "฿"
  },
  "TJR": {
    "displayName": "Tadschikistan Rubel",
    "displayName-count-one": "Tadschikistan-Rubel",
    "displayName-count-other": "Tadschikistan-Rubel",
    "symbol": "TJR"
  },
  "TJS": {
    "displayName": "Tadschikistan-Somoni",
    "displayName-count-one": "Tadschikistan-Somoni",
    "displayName-count-other": "Tadschikistan-Somoni",
    "symbol": "TJS"
  },
  "TMM": {
    "displayName": "Turkmenistan-Manat (1993-2009)",
    "displayName-count-one": "Turkmenistan-Manat (1993-2009)",
    "displayName-count-other": "Turkmenistan-Manat (1993-2009)",
    "symbol": "TMM"
  },
  "TMT": {
    "displayName": "Turkmenistan-Manat",
    "displayName-count-one": "Turkmenistan-Manat",

```

```

        "displayName-count-other": "Turkmenistan-Manat",
        "symbol": "TMT"
    },
    "TND": {
        "displayName": "Tunesischer Dinar",
        "displayName-count-one": "Tunesischer Dinar",
        "displayName-count-other": "Tunesische Dinar",
        "symbol": "TND"
    },
    "TOP": {
        "displayName": "Tongaischer Pa'anga",
        "displayName-count-one": "Tongaischer Pa'anga",
        "displayName-count-other": "Tongaische Pa'anga",
        "symbol": "TOP",
        "symbol-alt-narrow": "T$"
    },
    "TPE": {
        "displayName": "Timor-Escudo",
        "displayName-count-one": "Timor-Escudo",
        "displayName-count-other": "Timor-Escudo",
        "symbol": "TPE"
    },
    "TRL": {
        "displayName": "Türkische Lira (1922-2005)",
        "displayName-count-one": "Türkische Lira (1922-2005)",
        "displayName-count-other": "Türkische Lira (1922-2005)",
        "symbol": "TRL"
    },
    "TRY": {
        "displayName": "Türkische Lira",
        "displayName-count-one": "Türkische Lira",
        "displayName-count-other": "Türkische Lira",
        "symbol": "TRY",
        "symbol-alt-narrow": "₺",
        "symbol-alt-variant": "TL"
    },
    "TTD": {
        "displayName": "Trinidad und Tobago-Dollar",
        "displayName-count-one": "Trinidad und Tobago-Dollar",
        "displayName-count-other": "Trinidad und Tobago-Dollar",
        "symbol": "TTD",
        "symbol-alt-narrow": "$"
    },
    "TWD": {
        "displayName": "Neuer Taiwan-Dollar",
        "displayName-count-one": "Neuer Taiwan-Dollar",
        "displayName-count-other": "Neue Taiwan-Dollar",
        "symbol": "NT$",
        "symbol-alt-narrow": "NT$"
    },
    "TZS": {
        "displayName": "Tansania-Schilling",
        "displayName-count-one": "Tansania-Schilling",
        "displayName-count-other": "Tansania-Schilling",
        "symbol": "TZS"
    },
    "UAH": {

```

```

        "displayName": "Ukrainische Hrywnja",
        "displayName-count-one": "Ukrainische Hrywnja",
        "displayName-count-other": "Ukrainische Hrywen",
        "symbol": "UAH",
        "symbol-alt-narrow": "₴"
    },
    "UAK": {
        "displayName": "Ukrainischer Karbovanetz",
        "displayName-count-one": "Ukrainische Karbovanetz",
        "displayName-count-other": "Ukrainische Karbovanetz",
        "symbol": "UAK"
    },
    "UGS": {
        "displayName": "Uganda-Schilling (1966-1987)",
        "displayName-count-one": "Uganda-Schilling (1966-1987)",
        "displayName-count-other": "Uganda-Schilling (1966-1987)",
        "symbol": "UGS"
    },
    "UGX": {
        "displayName": "Uganda-Schilling",
        "displayName-count-one": "Uganda-Schilling",
        "displayName-count-other": "Uganda-Schilling",
        "symbol": "UGX"
    },
    "USD": {
        "displayName": "US-Dollar",
        "displayName-count-one": "US-Dollar",
        "displayName-count-other": "US-Dollar",
        "symbol": "$",
        "symbol-alt-narrow": "$"
    },
    "USN": {
        "displayName": "US Dollar (Nächster Tag)",
        "displayName-count-one": "US-Dollar (Nächster Tag)",
        "displayName-count-other": "US-Dollar (Nächster Tag)",
        "symbol": "USN"
    },
    "USS": {
        "displayName": "US Dollar (Gleicher Tag)",
        "displayName-count-one": "US-Dollar (Gleicher Tag)",
        "displayName-count-other": "US-Dollar (Gleicher Tag)",
        "symbol": "USS"
    },
    "UYI": {
        "displayName": "Uruguayischer Peso (Indexierte
Rechnungseinheiten)",
        "displayName-count-one": "Uruguayischer Peso (Indexierte
Rechnungseinheiten)",
        "displayName-count-other": "Uruguayische Pesos (Indexierte
Rechnungseinheiten)",
        "symbol": "UYI"
    },
    "UYP": {
        "displayName": "Uruguayischer Peso (1975-1993)",
        "displayName-count-one": "Uruguayischer Peso (1975-1993)",
        "displayName-count-other": "Uruguayische Pesos (1975-1993)",
        "symbol": "UYP"
    }

```

```

    },
    "UYU": {
      "displayName": "Uruguayischer Peso",
      "displayName-count-one": "Uruguayischer Peso",
      "displayName-count-other": "Uruguayische Pesos",
      "symbol": "UYU",
      "symbol-alt-narrow": "$"
    },
    "UZS": {
      "displayName": "Usbekistan-Sum",
      "displayName-count-one": "Usbekistan-Sum",
      "displayName-count-other": "Usbekistan-Sum",
      "symbol": "UZS"
    },
    "VEB": {
      "displayName": "Venezolanischer Bolívar (1871-2008)",
      "displayName-count-one": "Venezolanischer Bolívar (1871-2008)",
      "displayName-count-other": "Venezolanische Bolívares (1871-
2008)",
      "symbol": "VEB"
    },
    "VEF": {
      "displayName": "Venezolanischer Bolívar",
      "displayName-count-one": "Venezolanischer Bolívar",
      "displayName-count-other": "Venezolanische Bolívares",
      "symbol": "VEF",
      "symbol-alt-narrow": "Bs"
    },
    "VND": {
      "displayName": "Vietnamesischer Dong",
      "displayName-count-one": "Vietnamesischer Dong",
      "displayName-count-other": "Vietnamesische Dong",
      "symbol": "₫",
      "symbol-alt-narrow": "₫"
    },
    "VNN": {
      "displayName": "Vietnamesischer Dong(1978-1985)",
      "displayName-count-one": "Vietnamesischer Dong(1978-1985)",
      "displayName-count-other": "Vietnamesische Dong(1978-1985)",
      "symbol": "VNN"
    },
    "VUV": {
      "displayName": "Vanuatu-Vatu",
      "displayName-count-one": "Vanuatu-Vatu",
      "displayName-count-other": "Vanuatu-Vatu",
      "symbol": "VUV"
    },
    "WST": {
      "displayName": "Samoanischer Tala",
      "displayName-count-one": "Samoanischer Tala",
      "displayName-count-other": "Samoanische Tala",
      "symbol": "WST"
    },
    "XAF": {
      "displayName": "CFA-Franc (BEAC)",
      "displayName-count-one": "CFA-Franc (BEAC)",
      "displayName-count-other": "CFA-Franc (BEAC)",

```



```

        "symbol": "FCFA"
    },
    "XAG": {
        "displayName": "Unze Silber",
        "displayName-count-one": "Unze Silber",
        "displayName-count-other": "Unzen Silber",
        "symbol": "XAG"
    },
    "XAU": {
        "displayName": "Unze Gold",
        "displayName-count-one": "Unze Gold",
        "displayName-count-other": "Unzen Gold",
        "symbol": "XAU"
    },
    "XBA": {
        "displayName": "Europäische Rechnungseinheit",
        "displayName-count-one": "Europäische Rechnungseinheiten",
        "displayName-count-other": "Europäische Rechnungseinheiten",
        "symbol": "XBA"
    },
    (XBB) ",
    "XBB": {
        "displayName": "Europäische Währungseinheit (XBB)",
        "displayName-count-one": "Europäische Währungseinheiten (XBB)",
        "displayName-count-other": "Europäische Währungseinheiten",
        "symbol": "XBB"
    },
    "XBC": {
        "displayName": "Europäische Rechnungseinheit (XBC)",
        (XBC) ",
        "displayName-count-one": "Europäische Rechnungseinheiten (XBC)",
        "displayName-count-other": "Europäische Rechnungseinheiten",
        "symbol": "XBC"
    },
    "XBD": {
        "displayName": "Europäische Rechnungseinheit (XBD)",
        (XBD) ",
        "displayName-count-one": "Europäische Rechnungseinheiten (XBD)",
        "displayName-count-other": "Europäische Rechnungseinheiten",
        "symbol": "XBD"
    },
    "XCD": {
        "displayName": "Ostkaribischer Dollar",
        "displayName-count-one": "Ostkaribischer Dollar",
        "displayName-count-other": "Ostkaribische Dollar",
        "symbol": "EC$",
        "symbol-alt-narrow": "$"
    },
    "XDR": {
        "displayName": "Sonderziehungsrechte",
        "displayName-count-one": "Sonderziehungsrechte",
        "displayName-count-other": "Sonderziehungsrechte",
        "symbol": "XDR"
    },
    "XEU": {
        "displayName": "Europäische Währungseinheit (XEU)",
        "displayName-count-one": "Europäische Währungseinheiten (XEU)",

```

```

        "displayName-count-other": "Europäische Währungseinheiten
(XEU) ",
        "symbol": "XEU"
    },
    "XFO": {
        "displayName": "Französischer Gold-Franc",
        "displayName-count-one": "Französische Gold-Franc",
        "displayName-count-other": "Französische Gold-Franc",
        "symbol": "XFO"
    },
    "XFU": {
        "displayName": "Französischer UIC-Franc",
        "displayName-count-one": "Französische UIC-Franc",
        "displayName-count-other": "Französische UIC-Franc",
        "symbol": "XFU"
    },
    "XOF": {
        "displayName": "CFA-Franc (BCEAO) ",
        "displayName-count-one": "CFA-Franc (BCEAO) ",
        "displayName-count-other": "CFA-Francs (BCEAO) ",
        "symbol": "CFA"
    },
    "XPD": {
        "displayName": "Unze Palladium",
        "displayName-count-one": "Unze Palladium",
        "displayName-count-other": "Unzen Palladium",
        "symbol": "XPD"
    },
    "XPF": {
        "displayName": "CFP-Franc",
        "displayName-count-one": "CFP-Franc",
        "displayName-count-other": "CFP-Franc",
        "symbol": "CFPF"
    },
    "XPT": {
        "displayName": "Unze Platin",
        "displayName-count-one": "Unze Platin",
        "displayName-count-other": "Unzen Platin",
        "symbol": "XPT"
    },
    "XRE": {
        "displayName": "RINET Funds",
        "displayName-count-one": "RINET Funds",
        "displayName-count-other": "RINET Funds",
        "symbol": "XRE"
    },
    "XSU": {
        "displayName": "SUCRE",
        "displayName-count-one": "SUCRE",
        "displayName-count-other": "SUCRE",
        "symbol": "XSU"
    },
    "XTS": {
        "displayName": "Testwährung",
        "displayName-count-one": "Testwährung",
        "displayName-count-other": "Testwährung",
        "symbol": "XTS"
    }

```

```

    },
    "XUA": {
      "displayName": "Rechnungseinheit der AfEB",
      "displayName-count-one": "Rechnungseinheit der AfEB",
      "displayName-count-other": "Rechnungseinheiten der AfEB",
      "symbol": "XUA"
    },
    "XXX": {
      "displayName": "Unbekannte Währung",
      "displayName-count-one": "(unbekannte Währung)",
      "displayName-count-other": "(unbekannte Währung)",
      "symbol": "XXX"
    },
    "YDD": {
      "displayName": "Jemen-Dinar",
      "displayName-count-one": "Jemen-Dinar",
      "displayName-count-other": "Jemen-Dinar",
      "symbol": "YDD"
    },
    "YER": {
      "displayName": "Jemen-Rial",
      "displayName-count-one": "Jemen-Rial",
      "displayName-count-other": "Jemen-Rial",
      "symbol": "YER"
    },
    "YUD": {
      "displayName": "Jugoslawischer Dinar (1966-1990)",
      "displayName-count-one": "Jugoslawischer Dinar (1966-1990)",
      "displayName-count-other": "Jugoslawische Dinar (1966-1990)",
      "symbol": "YUD"
    },
    "YUM": {
      "displayName": "Jugoslawischer Neuer Dinar (1994-2002)",
      "displayName-count-one": "Jugoslawischer Neuer Dinar (1994-2002)",
      "displayName-count-other": "Jugoslawische Neue Dinar (1994-2002)",
      "symbol": "YUM"
    },
    "YUN": {
      "displayName": "Jugoslawischer Dinar (konvertibel)",
      "displayName-count-one": "Jugoslawische Dinar (konvertibel)",
      "displayName-count-other": "Jugoslawische Dinar (konvertibel)",
      "symbol": "YUN"
    },
    "YUR": {
      "displayName": "Jugoslawischer reformierter Dinar (1992-1993)",
      "displayName-count-one": "Jugoslawischer reformierter Dinar (1992-1993)",
      "displayName-count-other": "Jugoslawische reformierte Dinar (1992-1993)",
      "symbol": "YUR"
    },
    "ZAL": {
      "displayName": "Südafrikanischer Rand (Finanz)",
      "displayName-count-one": "Südafrikanischer Rand (Finanz)",
      "displayName-count-other": "Südafrikanischer Rand (Finanz)",

```

```

    "symbol": "ZAL"
  },
  "ZAR": {
    "displayName": "Südafrikanischer Rand",
    "displayName-count-one": "Südafrikanischer Rand",
    "displayName-count-other": "Südafrikanische Rand",
    "symbol": "ZAR",
    "symbol-alt-narrow": "R"
  },
  "ZMK": {
    "displayName": "Kwacha (1968-2012)",
    "displayName-count-one": "Kwacha (1968-2012)",
    "displayName-count-other": "Kwacha (1968-2012)",
    "symbol": "ZMK"
  },
  "ZMW": {
    "displayName": "Kwacha",
    "displayName-count-one": "Kwacha",
    "displayName-count-other": "Kwacha",
    "symbol": "ZMW",
    "symbol-alt-narrow": "K"
  },
  "ZRN": {
    "displayName": "Zaire-Neuer Zaïre (1993-1998)",
    "displayName-count-one": "Zaire-Neuer Zaïre (1993-1998)",
    "displayName-count-other": "Zaire-Neue Zaïre (1993-1998)",
    "symbol": "ZRN"
  },
  "ZRZ": {
    "displayName": "Zaire-Zaïre (1971-1993)",
    "displayName-count-one": "Zaire-Zaïre (1971-1993)",
    "displayName-count-other": "Zaire-Zaïre (1971-1993)",
    "symbol": "ZRZ"
  },
  "ZWD": {
    "displayName": "Simbabwe-Dollar (1980-2008)",
    "displayName-count-one": "Simbabwe-Dollar (1980-2008)",
    "displayName-count-other": "Simbabwe-Dollar (1980-2008)",
    "symbol": "ZWD"
  },
  "ZWL": {
    "displayName": "Simbabwe-Dollar (2009)",
    "displayName-count-one": "Simbabwe-Dollar (2009)",
    "displayName-count-other": "Simbabwe-Dollar (2009)",
    "symbol": "ZWL"
  },
  "ZWR": {
    "displayName": "Simbabwe-Dollar (2008)",
    "displayName-count-one": "Simbabwe-Dollar (2008)",
    "displayName-count-other": "Simbabwe-Dollar (2008)",
    "symbol": "ZWR"
  }
}
}
}
}
}

```

CURRENCIES.JSX

```

{
  "main";
  {
    "de";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 13259 $",
          "_cldrVersion";
          "31";
        }
        "language";
        "de";
      }
      "numbers";
      {
        "currencies";
        {
          "ADP";
          {
            "displayName";
            "Andorranische Pesete",
            "displayName-count-one";
            "Andorranische Pesete",
            "displayName-count-other";
            "Andorranische Peseten",
            "symbol";
            "ADP";
          }
          "AED";
          {
            "displayName";
            "VAE-Dirham",
            "displayName-count-one";
            "VAE-Dirham",
            "displayName-count-other";
            "VAE-Dirham",
            "symbol";
            "AED";
          }
          "AFA";
          {
            "displayName";
            "Afghanische Afghani (1927-2002)",
            "displayName-count-one";
            "Afghanische Afghani (1927-2002)",
            "displayName-count-other";
            "Afghanische Afghani (1927-2002)",
            "symbol";
            "AFA";
          }
        }
      }
    }
  }
}

```

```

    }
    "AFN";
    {
        "displayName";
        "Afghanischer Afghani",
        "displayName-count-one";
        "Afghanischer Afghani",
        "displayName-count-other";
        "Afghanische Afghani",
        "symbol";
        "AFN";
    }
    "ALK";
    {
        "displayName";
        "Albanischer Lek (1946-1965)",
        "displayName-count-one";
        "Albanischer Lek (1946-1965)",
        "displayName-count-other";
        "Albanische Lek (1946-1965)";
    }
    "ALL";
    {
        "displayName";
        "Albanischer Lek",
        "displayName-count-one";
        "Albanischer Lek",
        "displayName-count-other";
        "Albanische Lek",
        "symbol";
        "ALL";
    }
    "AMD";
    {
        "displayName";
        "Armenischer Dram",
        "displayName-count-one";
        "Armenischer Dram",
        "displayName-count-other";
        "Armenische Dram",
        "symbol";
        "AMD";
    }
    "ANG";
    {
        "displayName";
        "Niederländische-Antillen-Gulden",
        "displayName-count-one";
        "Niederländische-Antillen-Gulden",
        "displayName-count-other";
        "Niederländische-Antillen-Gulden",
        "symbol";
        "ANG";
    }
    "AOA";
    {
        "displayName";

```

```

        "Angolanischer Kwanza",
        "displayName-count-one";
        "Angolanischer Kwanza",
        "displayName-count-other";
        "Angolanische Kwanza",
        "symbol";
        "AOA",
        "symbol-alt-narrow";
        "Kz";
    }
    "AOK";
    {
        "displayName";
        "Angolanischer Kwanza (1977-1990)",
        "displayName-count-one";
        "Angolanischer Kwanza (1977-1990)",
        "displayName-count-other";
        "Angolanische Kwanza (1977-1990)",
        "symbol";
        "AOK";
    }
    "AON";
    {
        "displayName";
        "Angolanischer Neuer Kwanza (1990-2000)",
        "displayName-count-one";
        "Angolanischer Neuer Kwanza (1990-2000)",
        "displayName-count-other";
        "Angolanische Neue Kwanza (1990-2000)",
        "symbol";
        "AON";
    }
    "AOR";
    {
        "displayName";
        "Angolanischer Kwanza Reajustado (1995-1999)",
        "displayName-count-one";
        "Angolanischer Kwanza Reajustado (1995-1999)",
        "displayName-count-other";
        "Angolanische Kwanza Reajustado (1995-1999)",
        "symbol";
        "AOR";
    }
    "ARA";
    {
        "displayName";
        "Argentinischer Austral",
        "displayName-count-one";
        "Argentinischer Austral",
        "displayName-count-other";
        "Argentinische Austral",
        "symbol";
        "ARA";
    }
    "ARL";
    {
        "displayName";

```

```

        "Argentinischer Peso Ley (1970-1983)",
        "displayName-count-one";
        "Argentinischer Peso Ley (1970-1983)",
        "displayName-count-other";
        "Argentinische Pesos Ley (1970-1983)",
        "symbol";
        "ARL";
    }
    "ARM";
    {
        "displayName";
        "Argentinischer Peso (1881-1970)",
        "displayName-count-one";
        "Argentinischer Peso (1881-1970)",
        "displayName-count-other";
        "Argentinische Pesos (1881-1970)",
        "symbol";
        "ARM";
    }
    "ARP";
    {
        "displayName";
        "Argentinischer Peso (1983-1985)",
        "displayName-count-one";
        "Argentinischer Peso (1983-1985)",
        "displayName-count-other";
        "Argentinische Pesos (1983-1985)",
        "symbol";
        "ARP";
    }
    "ARS";
    {
        "displayName";
        "Argentinischer Peso",
        "displayName-count-one";
        "Argentinischer Peso",
        "displayName-count-other";
        "Argentinische Pesos",
        "symbol";
        "ARS",
        "symbol-alt-narrow";
        "$";
    }
    "ATS";
    {
        "displayName";
        "Österreichischer Schilling",
        "displayName-count-one";
        "Österreichischer Schilling",
        "displayName-count-other";
        "Österreichische Schilling",
        "symbol";
        "ÖS";
    }
    "AUD";
    {
        "displayName";

```



```

        "Australischer Dollar",
        "displayName-count-one";
        "Australischer Dollar",
        "displayName-count-other";
        "Australische Dollar",
        "symbol";
        "AU$",
        "symbol-alt-narrow";
        "$";
    }
    "AWG";
    {
        "displayName";
        "Aruba-Florin",
        "displayName-count-one";
        "Aruba-Florin",
        "displayName-count-other";
        "Aruba-Florin",
        "symbol";
        "AWG";
    }
    "AZM";
    {
        "displayName";
        "Aserbaidtschan-Manat (1993-2006)",
        "displayName-count-one";
        "Aserbaidtschan-Manat (1993-2006)",
        "displayName-count-other";
        "Aserbaidtschan-Manat (1993-2006)",
        "symbol";
        "AZM";
    }
    "AZN";
    {
        "displayName";
        "Aserbaidtschan-Manat",
        "displayName-count-one";
        "Aserbaidtschan-Manat",
        "displayName-count-other";
        "Aserbaidtschan-Manat",
        "symbol";
        "AZN";
    }
    "BAD";
    {
        "displayName";
        "Bosnien und Herzegowina Dinar (1992-1994)",
        "displayName-count-one";
        "Bosnien und Herzegowina Dinar (1992-1994)",
        "displayName-count-other";
        "Bosnien und Herzegowina Dinar (1992-1994)",
        "symbol";
        "BAD";
    }
    "BAM";
    {
        "displayName";

```

```

        "Bosnien und Herzegowina Konvertierbare Mark",
        "displayName-count-one";
        "Bosnien und Herzegowina Konvertierbare Mark",
        "displayName-count-other";
        "Bosnien und Herzegowina Konvertierbare Mark",
        "symbol";
        "BAM",
        "symbol-alt-narrow";
        "KM";
    }
    "BAN";
    {
        "displayName";
        "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
        "displayName-count-one";
        "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
        "displayName-count-other";
        "Bosnien und Herzegowina Neue Dinar (1994-1997)",
        "symbol";
        "BAN";
    }
    "BBD";
    {
        "displayName";
        "Barbados-Dollar",
        "displayName-count-one";
        "Barbados-Dollar",
        "displayName-count-other";
        "Barbados-Dollar",
        "symbol";
        "BBD",
        "symbol-alt-narrow";
        "$";
    }
    "BDT";
    {
        "displayName";
        "Bangladesch-Taka",
        "displayName-count-one";
        "Bangladesch-Taka",
        "displayName-count-other";
        "Bangladesch-Taka",
        "symbol";
        "BDT",
        "symbol-alt-narrow";
        "৳";
    }
    "BEC";
    {
        "displayName";
        "Belgischer Franc (konvertibel)",
        "displayName-count-one";
        "Belgischer Franc (konvertibel)",
        "displayName-count-other";
        "Belgische Franc (konvertibel)",
        "symbol";
    }

```

```

        "BEC";
    }
    "BEF";
    {
        "displayName";
        "Belgischer Franc",
        "displayName-count-one";
        "Belgischer Franc",
        "displayName-count-other";
        "Belgische Franc",
        "symbol";
        "BEF";
    }
    "BEL";
    {
        "displayName";
        "Belgischer Finanz-Franc",
        "displayName-count-one";
        "Belgischer Finanz-Franc",
        "displayName-count-other";
        "Belgische Finanz-Franc",
        "symbol";
        "BEL";
    }
    "BGL";
    {
        "displayName";
        "Bulgarische Lew (1962-1999)",
        "displayName-count-one";
        "Bulgarische Lew (1962-1999)",
        "displayName-count-other";
        "Bulgarische Lew (1962-1999)",
        "symbol";
        "BGL";
    }
    "BGM";
    {
        "displayName";
        "Bulgarischer Lew (1952-1962)",
        "displayName-count-one";
        "Bulgarischer Lew (1952-1962)",
        "displayName-count-other";
        "Bulgarische Lew (1952-1962)",
        "symbol";
        "BGK";
    }
    "BGN";
    {
        "displayName";
        "Bulgarischer Lew",
        "displayName-count-one";
        "Bulgarischer Lew",
        "displayName-count-other";
        "Bulgarische Lew",
        "symbol";
        "BGN";
    }
}

```

```

"BGO";
{
  "displayName";
  "Bulgarischer Lew (1879-1952)",
    "displayName-count-one";
  "Bulgarischer Lew (1879-1952)",
    "displayName-count-other";
  "Bulgarische Lew (1879-1952)",
    "symbol";
  "BGJ";
}
"BHD";
{
  "displayName";
  "Bahrain-Dinar",
    "displayName-count-one";
  "Bahrain-Dinar",
    "displayName-count-other";
  "Bahrain-Dinar",
    "symbol";
  "BHD";
}
"BIF";
{
  "displayName";
  "Burundi-Franc",
    "displayName-count-one";
  "Burundi-Franc",
    "displayName-count-other";
  "Burundi-Francs",
    "symbol";
  "BIF";
}
"BMD";
{
  "displayName";
  "Bermuda-Dollar",
    "displayName-count-one";
  "Bermuda-Dollar",
    "displayName-count-other";
  "Bermuda-Dollar",
    "symbol";
  "BMD",
    "symbol-alt-narrow";
  "$";
}
"BND";
{
  "displayName";
  "Brunei-Dollar",
    "displayName-count-one";
  "Brunei-Dollar",
    "displayName-count-other";
  "Brunei-Dollar",
    "symbol";
  "BND",
    "symbol-alt-narrow";
}

```

```

        "$";
    }
    "BOB";
    {
        "displayName";
        "Bolivanischer Boliviano",
        "displayName-count-one";
        "Bolivanischer Boliviano",
        "displayName-count-other";
        "Bolivianische Bolivianos",
        "symbol";
        "BOB",
        "symbol-alt-narrow";
        "Bs";
    }
    "BOL";
    {
        "displayName";
        "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-one";
        "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-other";
        "Bolivianische Bolivianos (1863-1963)",
        "symbol";
        "BOL";
    }
    "BOP";
    {
        "displayName";
        "Bolivianischer Peso",
        "displayName-count-one";
        "Bolivianischer Peso",
        "displayName-count-other";
        "Bolivianische Peso",
        "symbol";
        "BOP";
    }
    "BOV";
    {
        "displayName";
        "Boliviansiche Mvdol",
        "displayName-count-one";
        "Boliviansiche Mvdol",
        "displayName-count-other";
        "Bolivianische Mvdol",
        "symbol";
        "BOV";
    }
    "BRB";
    {
        "displayName";
        "Brasilianischer Cruzeiro Novo (1967-1986)",
        "displayName-count-one";
        "Brasilianischer Cruzeiro Novo (1967-1986)",
        "displayName-count-other";
        "Brasilianische Cruzeiro Novo (1967-1986)",
        "symbol";
    }

```

```

        "BRB";
    }
    "BRC";
    {
        "displayName";
        "Brasilianischer Cruzado (1986-1989)",
        "displayName-count-one";
        "Brasilianischer Cruzado (1986-1989)",
        "displayName-count-other";
        "Brasilianische Cruzado (1986-1989)",
        "symbol";
        "BRC";
    }
    "BRE";
    {
        "displayName";
        "Brasilianischer Cruzeiro (1990-1993)",
        "displayName-count-one";
        "Brasilianischer Cruzeiro (1990-1993)",
        "displayName-count-other";
        "Brasilianische Cruzeiro (1990-1993)",
        "symbol";
        "BRE";
    }
    "BRL";
    {
        "displayName";
        "Brasilianischer Real",
        "displayName-count-one";
        "Brasilianischer Real",
        "displayName-count-other";
        "Brasilianische Real",
        "symbol";
        "R$",
        "symbol-alt-narrow";
        "R$";
    }
    "BRN";
    {
        "displayName";
        "Brasilianischer Cruzado Novo (1989-1990)",
        "displayName-count-one";
        "Brasilianischer Cruzado Novo (1989-1990)",
        "displayName-count-other";
        "Brasilianische Cruzado Novo (1989-1990)",
        "symbol";
        "BRN";
    }
    "BRR";
    {
        "displayName";
        "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-one";
        "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-other";
        "Brasilianische Cruzeiro (1993-1994)",
        "symbol";
    }

```

```

        "BRR";
    }
    "BRZ";
    {
        "displayName";
        "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-one";
        "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-other";
        "Brasilianischer Cruzeiro (1942-1967)",
        "symbol";
        "BRZ";
    }
    "BSD";
    {
        "displayName";
        "Bahamas-Dollar",
        "displayName-count-one";
        "Bahamas-Dollar",
        "displayName-count-other";
        "Bahamas-Dollar",
        "symbol";
        "BSD",
        "symbol-alt-narrow";
        "$";
    }
    "BTN";
    {
        "displayName";
        "Bhutan-Ngultrum",
        "displayName-count-one";
        "Bhutan-Ngultrum",
        "displayName-count-other";
        "Bhutan-Ngultrum",
        "symbol";
        "BTN";
    }
    "BUK";
    {
        "displayName";
        "Birmanischer Kyat",
        "displayName-count-one";
        "Birmanischer Kyat",
        "displayName-count-other";
        "Birmanische Kyat",
        "symbol";
        "BUK";
    }
    "BWP";
    {
        "displayName";
        "Botswanischer Pula",
        "displayName-count-one";
        "Botswanischer Pula",
        "displayName-count-other";
        "Botswanische Pula",
        "symbol";
    }

```

```

        "BWP",
        "symbol-alt-narrow";
        "P";
    }
    "BYB";
    {
        "displayName";
        "Belarus-Rubel (1994-1999)",
        "displayName-count-one";
        "Belarus-Rubel (1994-1999)",
        "displayName-count-other";
        "Belarus-Rubel (1994-1999)",
        "symbol";
        "BYB";
    }
    "BYN";
    {
        "displayName";
        "Weißrussischer Rubel",
        "displayName-count-one";
        "Weißrussischer Rubel",
        "displayName-count-other";
        "Weißrussische Rubel",
        "symbol";
        "BYN",
        "symbol-alt-narrow";
        "p.";
    }
    "BYR";
    {
        "displayName";
        "Weißrussischer Rubel (2000-2016)",
        "displayName-count-one";
        "Weißrussischer Rubel (2000-2016)",
        "displayName-count-other";
        "Weißrussische Rubel (2000-2016)",
        "symbol";
        "BYR";
    }
    "BZD";
    {
        "displayName";
        "Belize-Dollar",
        "displayName-count-one";
        "Belize-Dollar",
        "displayName-count-other";
        "Belize-Dollar",
        "symbol";
        "BZD",
        "symbol-alt-narrow";
        "$";
    }
    "CAD";
    {
        "displayName";
        "Kanadischer Dollar",
        "displayName-count-one";

```



```

        "Kanadischer Dollar",
        "displayName-count-other";
    "Kanadische Dollar",
        "symbol";
    "CA$",
        "symbol-alt-narrow";
    "$";
}
"CDF";
{
    "displayName";
    "Kongo-Franc",
        "displayName-count-one";
    "Kongo-Franc",
        "displayName-count-other";
    "Kongo-Francs",
        "symbol";
    "CDF";
}
"CHE";
{
    "displayName";
    "WIR-Euro",
        "displayName-count-one";
    "WIR-Euro",
        "displayName-count-other";
    "WIR-Euro",
        "symbol";
    "CHE";
}
"CHF";
{
    "displayName";
    "Schweizer Franken",
        "displayName-count-one";
    "Schweizer Franken",
        "displayName-count-other";
    "Schweizer Franken",
        "symbol";
    "CHF";
}
"CHW";
{
    "displayName";
    "WIR Franken",
        "displayName-count-one";
    "WIR Franken",
        "displayName-count-other";
    "WIR Franken",
        "symbol";
    "CHW";
}
"CLE";
{
    "displayName";
    "Chilenischer Escudo",
        "displayName-count-one";

```

```

        "Chilenischer Escudo",
        "displayName-count-other";
    "Chilenische Escudo",
        "symbol";
    "CLE";
}
"CLF";
{
    "displayName";
    "Chilenische Unidades de Fomento",
        "displayName-count-one";
    "Chilenische Unidades de Fomento",
        "displayName-count-other";
    "Chilenische Unidades de Fomento",
        "symbol";
    "CLF";
}
"CLP";
{
    "displayName";
    "Chilenischer Peso",
        "displayName-count-one";
    "Chilenischer Peso",
        "displayName-count-other";
    "Chilenische Pesos",
        "symbol";
    "CLP",
        "symbol-alt-narrow";
    "$";
}
"CNX";
{
    "displayName";
    "Dollar der Chinesischen Volksbank",
        "displayName-count-one";
    "Dollar der Chinesischen Volksbank",
        "displayName-count-other";
    "Dollar der Chinesischen Volksbank",
        "symbol";
    "CNX";
}
"CNY";
{
    "displayName";
    "Renminbi Yuan",
        "displayName-count-one";
    "Chinesischer Yuan",
        "displayName-count-other";
    "Renminbi Yuan",
        "symbol";
    "CN¥",
        "symbol-alt-narrow";
    "¥";
}
"COP";
{
    "displayName";

```

```

        "Kolumbianischer Peso",
        "displayName-count-one";
        "Kolumbianischer Peso",
        "displayName-count-other";
        "Kolumbianische Pesos",
        "symbol";
        "COP",
        "symbol-alt-narrow";
        "$";
    }
    "COU";
    {
        "displayName";
        "Kolumbianische Unidades de valor real",
        "displayName-count-one";
        "Kolumbianische Unidad de valor real",
        "displayName-count-other";
        "Kolumbianische Unidades de valor real",
        "symbol";
        "COU";
    }
    "CRC";
    {
        "displayName";
        "Costa-Rica-Colón",
        "displayName-count-one";
        "Costa-Rica-Colón",
        "displayName-count-other";
        "Costa-Rica-Colón",
        "symbol";
        "CRC",
        "symbol-alt-narrow";
        "₡";
    }
    "CSD";
    {
        "displayName";
        "Serbischer Dinar (2002-2006)",
        "displayName-count-one";
        "Serbischer Dinar (2002-2006)",
        "displayName-count-other";
        "Serbische Dinar (2002-2006)",
        "symbol";
        "CSD";
    }
    "CSK";
    {
        "displayName";
        "Tschechoslowakische Krone",
        "displayName-count-one";
        "Tschechoslowakische Kronen",
        "displayName-count-other";
        "Tschechoslowakische Kronen",
        "symbol";
        "CSK";
    }
    "CUC";

```

```

    {
      "displayName";
      "Kubanischer Peso (konvertibel)",
        "displayName-count-one";
      "Kubanischer Peso (konvertibel)",
        "displayName-count-other";
      "Kubanische Pesos (konvertibel)",
        "symbol";
      "CUC",
        "symbol-alt-narrow";
      "Cub$";
    }
    "CUP";
    {
      "displayName";
      "Kubanischer Peso",
        "displayName-count-one";
      "Kubanischer Peso",
        "displayName-count-other";
      "Kubanische Pesos",
        "symbol";
      "CUP",
        "symbol-alt-narrow";
      "$";
    }
    "CVE";
    {
      "displayName";
      "Cabo-Verde-Escudo",
        "displayName-count-one";
      "Cabo-Verde-Escudo",
        "displayName-count-other";
      "Cabo-Verde-Escudos",
        "symbol";
      "CVE";
    }
    "CYP";
    {
      "displayName";
      "Zypern-Pfund",
        "displayName-count-one";
      "Zypern Pfund",
        "displayName-count-other";
      "Zypern Pfund",
        "symbol";
      "CYP";
    }
    "CZK";
    {
      "displayName";
      "Tschechische Krone",
        "displayName-count-one";
      "Tschechische Krone",
        "displayName-count-other";
      "Tschechische Kronen",
        "symbol";
      "CZK",

```

```

        "symbol-alt-narrow";
        "Kč";
    }
    "DDM";
    {
        "displayName";
        "Mark der DDR",
        "displayName-count-one";
        "Mark der DDR",
        "displayName-count-other";
        "Mark der DDR",
        "symbol";
        "DDM";
    }
    "DEM";
    {
        "displayName";
        "Deutsche Mark",
        "displayName-count-one";
        "Deutsche Mark",
        "displayName-count-other";
        "Deutsche Mark",
        "symbol";
        "DM";
    }
    "DJF";
    {
        "displayName";
        "Dschibuti-Franc",
        "displayName-count-one";
        "Dschibuti-Franc",
        "displayName-count-other";
        "Dschibuti-Franc",
        "symbol";
        "DJF";
    }
    "DKK";
    {
        "displayName";
        "Dänische Krone",
        "displayName-count-one";
        "Dänische Krone",
        "displayName-count-other";
        "Dänische Kronen",
        "symbol";
        "DKK",
        "symbol-alt-narrow";
        "kr";
    }
    "DOP";
    {
        "displayName";
        "Dominikanischer Peso",
        "displayName-count-one";
        "Dominikanischer Peso",
        "displayName-count-other";
        "Dominikanische Pesos",

```

```

        "symbol";
        "DOP",
        "symbol-alt-narrow";
        "$";
    }
    "DZD";
    {
        "displayName";
        "Algerischer Dinar",
        "displayName-count-one";
        "Algerischer Dinar",
        "displayName-count-other";
        "Algerische Dinar",
        "symbol";
        "DZD";
    }
    "ECS";
    {
        "displayName";
        "Ecuadorianischer Sucre",
        "displayName-count-one";
        "Ecuadorianischer Sucre",
        "displayName-count-other";
        "Ecuadorianische Sucre",
        "symbol";
        "ECS";
    }
    "ECV";
    {
        "displayName";
        "Verrechnungseinheit für Ecuador",
        "displayName-count-one";
        "Verrechnungseinheiten für Ecuador",
        "displayName-count-other";
        "Verrechnungseinheiten für Ecuador",
        "symbol";
        "ECV";
    }
    "EEK";
    {
        "displayName";
        "Estnische Krone",
        "displayName-count-one";
        "Estnische Krone",
        "displayName-count-other";
        "Estnische Kronen",
        "symbol";
        "EEK";
    }
    "EGP";
    {
        "displayName";
        "Ägyptisches Pfund",
        "displayName-count-one";
        "Ägyptisches Pfund",
        "displayName-count-other";
        "Ägyptische Pfund",

```

```

        "symbol";
        "EGP",
        "symbol-alt-narrow";
        "£";
    }
    "ERN";
    {
        "displayName";
        "Eritreischer Nakfa",
        "displayName-count-one";
        "Eritreischer Nakfa",
        "displayName-count-other";
        "Eritreische Nakfa",
        "symbol";
        "ERN";
    }
    "ESA";
    {
        "displayName";
        "Spanische Peseta (A-Konten)",
        "displayName-count-one";
        "Spanische Peseta (A-Konten)",
        "displayName-count-other";
        "Spanische Peseten (A-Konten)",
        "symbol";
        "ESA";
    }
    "ESB";
    {
        "displayName";
        "Spanische Peseta (konvertibel)",
        "displayName-count-one";
        "Spanische Peseta (konvertibel)",
        "displayName-count-other";
        "Spanische Peseten (konvertibel)",
        "symbol";
        "ESB";
    }
    "ESP";
    {
        "displayName";
        "Spanische Peseta",
        "displayName-count-one";
        "Spanische Peseta",
        "displayName-count-other";
        "Spanische Peseten",
        "symbol";
        "ESP",
        "symbol-alt-narrow";
        "₧";
    }
    "ETB";
    {
        "displayName";
        "Äthiopischer Birr",
        "displayName-count-one";
        "Äthiopischer Birr",

```

```

        "displayName-count-other";
        "Äthiopische Birr",
        "symbol";
        "ETB";
    }
    "EUR";
    {
        "displayName";
        "Euro",
        "displayName-count-one";
        "Euro",
        "displayName-count-other";
        "Euro",
        "symbol";
        "€",
        "symbol-alt-narrow";
        "€";
    }
    "FIM";
    {
        "displayName";
        "Finnische Mark",
        "displayName-count-one";
        "Finnische Mark",
        "displayName-count-other";
        "Finnische Mark",
        "symbol";
        "FIM";
    }
    "FJD";
    {
        "displayName";
        "Fidschi-Dollar",
        "displayName-count-one";
        "Fidschi-Dollar",
        "displayName-count-other";
        "Fidschi-Dollar",
        "symbol";
        "FJD",
        "symbol-alt-narrow";
        "$";
    }
    "FKP";
    {
        "displayName";
        "Falkland-Pfund",
        "displayName-count-one";
        "Falkland-Pfund",
        "displayName-count-other";
        "Falkland-Pfund",
        "symbol";
        "FKP",
        "symbol-alt-narrow";
        "Fl£";
    }
    "FRF";
    {

```



```

        "displayName";
        "Französischer Franc",
        "displayName-count-one";
        "Französischer Franc",
        "displayName-count-other";
        "Französische Franc",
        "symbol";
        "FRF";
    }
    "GBP";
    {
        "displayName";
        "Britisches Pfund",
        "displayName-count-one";
        "Britisches Pfund",
        "displayName-count-other";
        "Britische Pfund",
        "symbol";
        "£",
        "symbol-alt-narrow";
        "£";
    }
    "GEK";
    {
        "displayName";
        "Georgischer Kupon Larit",
        "displayName-count-one";
        "Georgischer Kupon Larit",
        "displayName-count-other";
        "Georgische Kupon Larit",
        "symbol";
        "GEK";
    }
    "GEL";
    {
        "displayName";
        "Georgischer Lari",
        "displayName-count-one";
        "Georgischer Lari",
        "displayName-count-other";
        "Georgische Lari",
        "symbol";
        "GEL",
        "symbol-alt-narrow";
        "ლ",
        "symbol-alt-variant";
        "ლ";
    }
    "GHC";
    {
        "displayName";
        "Ghanaischer Cedi (1979-2007)",
        "displayName-count-one";
        "Ghanaischer Cedi (1979-2007)",
        "displayName-count-other";
        "Ghanaische Cedi (1979-2007)",
        "symbol";
    }

```

```

        "GHC";
    }
    "GHS";
    {
        "displayName";
        "Ghanaischer Cedi",
        "displayName-count-one";
        "Ghanaischer Cedi",
        "displayName-count-other";
        "Ghanaische Cedi",
        "symbol";
        "GHS";
    }
    "GIP";
    {
        "displayName";
        "Gibraltar-Pfund",
        "displayName-count-one";
        "Gibraltar-Pfund",
        "displayName-count-other";
        "Gibraltar Pfund",
        "symbol";
        "GIP",
        "symbol-alt-narrow";
        "£";
    }
    "GMD";
    {
        "displayName";
        "Gambia-Dalasi",
        "displayName-count-one";
        "Gambia-Dalasi",
        "displayName-count-other";
        "Gambia-Dalasi",
        "symbol";
        "GMD";
    }
    "GNF";
    {
        "displayName";
        "Guinea-Franc",
        "displayName-count-one";
        "Guinea-Franc",
        "displayName-count-other";
        "Guinea-Franc",
        "symbol";
        "GNF",
        "symbol-alt-narrow";
        "F.G.";
    }
    "GNS";
    {
        "displayName";
        "Guineischer Syli",
        "displayName-count-one";
        "Guineischer Syli",
        "displayName-count-other";
    }

```

```

        "Guineische Syli",
        "symbol";
        "GNS";
    }
    "GQE";
    {
        "displayName";
        "Äquatorialguinea-Ekwele",
        "displayName-count-one";
        "Äquatorialguinea-Ekwele",
        "displayName-count-other";
        "Äquatorialguinea-Ekwele",
        "symbol";
        "GQE";
    }
    "GRD";
    {
        "displayName";
        "Griechische Drachme",
        "displayName-count-one";
        "Griechische Drachme",
        "displayName-count-other";
        "Griechische Drachmen",
        "symbol";
        "GRD";
    }
    "GTQ";
    {
        "displayName";
        "Guatemaltekinscher Quetzal",
        "displayName-count-one";
        "Guatemaltekinscher Quetzal",
        "displayName-count-other";
        "Guatemaltekinsche Quetzales",
        "symbol";
        "GTQ",
        "symbol-alt-narrow";
        "Q";
    }
    "GWE";
    {
        "displayName";
        "Portugiesisch Guinea Escudo",
        "displayName-count-one";
        "Portugiesisch Guinea Escudo",
        "displayName-count-other";
        "Portugiesisch Guinea Escudo",
        "symbol";
        "GWE";
    }
    "GWP";
    {
        "displayName";
        "Guinea-Bissau Peso",
        "displayName-count-one";
        "Guinea-Bissau Peso",
        "displayName-count-other";
    }

```

```

        "Guinea-Bissau Pesos",
        "symbol";
        "GWP";
    }
    "GYD";
    {
        "displayName";
        "Guyana-Dollar",
        "displayName-count-one";
        "Guyana-Dollar",
        "displayName-count-other";
        "Guyana-Dollar",
        "symbol";
        "GYD",
        "symbol-alt-narrow";
        "$";
    }
    "HKD";
    {
        "displayName";
        "Hongkong-Dollar",
        "displayName-count-one";
        "Hongkong-Dollar",
        "displayName-count-other";
        "Hongkong-Dollar",
        "symbol";
        "HK$",
        "symbol-alt-narrow";
        "$";
    }
    "HNL";
    {
        "displayName";
        "Honduras-Lempira",
        "displayName-count-one";
        "Honduras-Lempira",
        "displayName-count-other";
        "Honduras-Lempira",
        "symbol";
        "HNL",
        "symbol-alt-narrow";
        "L";
    }
    "HRD";
    {
        "displayName";
        "Kroatischer Dinar",
        "displayName-count-one";
        "Kroatischer Dinar",
        "displayName-count-other";
        "Kroatische Dinar",
        "symbol";
        "HRD";
    }
    "HRK";
    {
        "displayName";

```

```

        "Kroatischer Kuna",
        "displayName-count-one";
        "Kroatischer Kuna",
        "displayName-count-other";
        "Kroatische Kuna",
        "symbol";
        "HRK",
        "symbol-alt-narrow";
        "kn";
    }
    "HTG";
    {
        "displayName";
        "Haitianische Gourde",
        "displayName-count-one";
        "Haitianische Gourde",
        "displayName-count-other";
        "Haitianische Gourdes",
        "symbol";
        "HTG";
    }
    "HUF";
    {
        "displayName";
        "Ungarischer Forint",
        "displayName-count-one";
        "Ungarischer Forint",
        "displayName-count-other";
        "Ungarische Forint",
        "symbol";
        "HUF",
        "symbol-alt-narrow";
        "Ft";
    }
    "IDR";
    {
        "displayName";
        "Indonesische Rupiah",
        "displayName-count-one";
        "Indonesische Rupiah",
        "displayName-count-other";
        "Indonesische Rupiah",
        "symbol";
        "IDR",
        "symbol-alt-narrow";
        "Rp";
    }
    "IEP";
    {
        "displayName";
        "Irisches Pfund",
        "displayName-count-one";
        "Irisches Pfund",
        "displayName-count-other";
        "Irische Pfund",
        "symbol";
        "IEP";
    }

```

```

    }
    "ILP";
    {
        "displayName";
        "Israelisches Pfund",
        "displayName-count-one";
        "Israelisches Pfund",
        "displayName-count-other";
        "Israelische Pfund",
        "symbol";
        "ILP";
    }
    "ILR";
    {
        "displayName";
        "Israelischer Schekel (1980-1985)",
        "displayName-count-one";
        "Israelischer Schekel (1980-1985)",
        "displayName-count-other";
        "Israelische Schekel (1980-1985)";
    }
    "ILS";
    {
        "displayName";
        "Israelischer Neuer Schekel",
        "displayName-count-one";
        "Israelischer Neuer Schekel",
        "displayName-count-other";
        "Israelische Neue Schekel",
        "symbol";
        "₪",
        "symbol-alt-narrow";
        "₪";
    }
    "INR";
    {
        "displayName";
        "Indische Rupie",
        "displayName-count-one";
        "Indische Rupie",
        "displayName-count-other";
        "Indische Rupien",
        "symbol";
        "₹",
        "symbol-alt-narrow";
        "₹";
    }
    "IQD";
    {
        "displayName";
        "Irakischer Dinar",
        "displayName-count-one";
        "Irakischer Dinar",
        "displayName-count-other";
        "Irakische Dinar",
        "symbol";
        "IQD";
    }

```

```

    }
    "IRR";
    {
        "displayName";
        "Iranischer Rial",
            "displayName-count-one";
        "Iranischer Rial",
            "displayName-count-other";
        "Iranische Rial",
            "symbol";
        "IRR";
    }
    "ISJ";
    {
        "displayName";
        "Isländische Krone (1918-1981)",
            "displayName-count-one";
        "Isländische Krone (1918-1981)",
            "displayName-count-other";
        "Isländische Kronen (1918-1981)";
    }
    "ISK";
    {
        "displayName";
        "Isländische Krone",
            "displayName-count-one";
        "Isländische Krone",
            "displayName-count-other";
        "Isländische Kronen",
            "symbol";
        "ISK",
            "symbol-alt-narrow";
        "kr";
    }
    "ITL";
    {
        "displayName";
        "Italienische Lira",
            "displayName-count-one";
        "Italienische Lira",
            "displayName-count-other";
        "Italienische Lire",
            "symbol";
        "ITL";
    }
    "JMD";
    {
        "displayName";
        "Jamaika-Dollar",
            "displayName-count-one";
        "Jamaika-Dollar",
            "displayName-count-other";
        "Jamaika-Dollar",
            "symbol";
        "JMD",
            "symbol-alt-narrow";
        "$";
    }

```

```

    }
    "JOD";
    {
        "displayName";
        "Jordanischer Dinar",
        "displayName-count-one";
        "Jordanischer Dinar",
        "displayName-count-other";
        "Jordanische Dinar",
        "symbol";
        "JOD";
    }
    "JPY";
    {
        "displayName";
        "Japanischer Yen",
        "displayName-count-one";
        "Japanischer Yen",
        "displayName-count-other";
        "Japanische Yen",
        "symbol";
        "¥",
        "symbol-alt-narrow";
        "¥";
    }
    "KES";
    {
        "displayName";
        "Kenia-Schilling",
        "displayName-count-one";
        "Kenia-Schilling",
        "displayName-count-other";
        "Kenia-Schilling",
        "symbol";
        "KES";
    }
    "KGS";
    {
        "displayName";
        "Kirgisischer Som",
        "displayName-count-one";
        "Kirgisischer Som",
        "displayName-count-other";
        "Kirgisische Som",
        "symbol";
        "KGS";
    }
    "KHR";
    {
        "displayName";
        "Kambodschanischer Riel",
        "displayName-count-one";
        "Kambodschanischer Riel",
        "displayName-count-other";
        "Kambodschanische Riel",
        "symbol";
        "KHR",

```



```

        "symbol-alt-narrow";
        "₭";
    }
    "KMF";
    {
        "displayName";
        "Komoren-Franc",
        "displayName-count-one";
        "Komoren-Franc",
        "displayName-count-other";
        "Komoren-Francs",
        "symbol";
        "KMF",
        "symbol-alt-narrow";
        "FC";
    }
    "KPW";
    {
        "displayName";
        "Nordkoreanischer Won",
        "displayName-count-one";
        "Nordkoreanischer Won",
        "displayName-count-other";
        "Nordkoreanische Won",
        "symbol";
        "KPW",
        "symbol-alt-narrow";
        "₩";
    }
    "KRH";
    {
        "displayName";
        "Südkoreanischer Hwan (1953-1962)",
        "displayName-count-one";
        "Südkoreanischer Hwan (1953-1962)",
        "displayName-count-other";
        "Südkoreanischer Hwan (1953-1962)",
        "symbol";
        "KRH";
    }
    "KRO";
    {
        "displayName";
        "Südkoreanischer Won (1945-1953)",
        "displayName-count-one";
        "Südkoreanischer Won (1945-1953)",
        "displayName-count-other";
        "Südkoreanischer Won (1945-1953)",
        "symbol";
        "KRO";
    }
    "KRW";
    {
        "displayName";
        "Südkoreanischer Won",
        "displayName-count-one";

```

```

        "Südkoreanischer Won",
        "displayName-count-other";
    "Südkoreanische Won",
        "symbol";
    "₩",
        "symbol-alt-narrow";
    "₩";
}
"KWD";
{
    "displayName";
    "Kuwait-Dinar",
        "displayName-count-one";
    "Kuwait-Dinar",
        "displayName-count-other";
    "Kuwait-Dinar",
        "symbol";
    "KWD";
}
"KYD";
{
    "displayName";
    "Kaiman-Dollar",
        "displayName-count-one";
    "Kaiman-Dollar",
        "displayName-count-other";
    "Kaiman-Dollar",
        "symbol";
    "KYD",
        "symbol-alt-narrow";
    "$";
}
"KZT";
{
    "displayName";
    "Kasachischer Tenge",
        "displayName-count-one";
    "Kasachischer Tenge",
        "displayName-count-other";
    "Kasachische Tenge",
        "symbol";
    "KZT",
        "symbol-alt-narrow";
    "₸";
}
"LAK";
{
    "displayName";
    "Laotischer Kip",
        "displayName-count-one";
    "Laotischer Kip",
        "displayName-count-other";
    "Laotische Kip",
        "symbol";
    "LAK",
        "symbol-alt-narrow";
    "₭";
}

```

```

    }
    "LBP";
    {
        "displayName";
        "Libanesisches Pfund",
        "displayName-count-one";
        "Libanesisches Pfund",
        "displayName-count-other";
        "Libanesische Pfund",
        "symbol";
        "LBP",
        "symbol-alt-narrow";
        "L£";
    }
    "LKR";
    {
        "displayName";
        "Sri-Lanka-Rupie",
        "displayName-count-one";
        "Sri-Lanka-Rupie",
        "displayName-count-other";
        "Sri-Lanka-Rupien",
        "symbol";
        "LKR",
        "symbol-alt-narrow";
        "Rs";
    }
    "LRD";
    {
        "displayName";
        "Liberianischer Dollar",
        "displayName-count-one";
        "Liberianischer Dollar",
        "displayName-count-other";
        "Liberianische Dollar",
        "symbol";
        "LRD",
        "symbol-alt-narrow";
        "$";
    }
    "LSL";
    {
        "displayName";
        "Loti",
        "displayName-count-one";
        "Loti",
        "displayName-count-other";
        "Loti",
        "symbol";
        "LSL";
    }
    "LTL";
    {
        "displayName";
        "Litauischer Litas",
        "displayName-count-one";
        "Litauischer Litas",

```

```

        "displayName-count-other";
        "Litauische Litas",
        "symbol";
        "LTL",
        "symbol-alt-narrow";
        "Lt";
    }
    "LTT";
    {
        "displayName";
        "Litauischer Talonas",
        "displayName-count-one";
        "Litauische Talonas",
        "displayName-count-other";
        "Litauische Talonas",
        "symbol";
        "LTT";
    }
    "LUC";
    {
        "displayName";
        "Luxemburgischer Franc (konvertibel)",
        "displayName-count-one";
        "Luxemburgische Franc (konvertibel)",
        "displayName-count-other";
        "Luxemburgische Franc (konvertibel)",
        "symbol";
        "LUC";
    }
    "LUF";
    {
        "displayName";
        "Luxemburgischer Franc",
        "displayName-count-one";
        "Luxemburgische Franc",
        "displayName-count-other";
        "Luxemburgische Franc",
        "symbol";
        "LUF";
    }
    "LUL";
    {
        "displayName";
        "Luxemburgischer Finanz-Franc",
        "displayName-count-one";
        "Luxemburgische Finanz-Franc",
        "displayName-count-other";
        "Luxemburgische Finanz-Franc",
        "symbol";
        "LUL";
    }
    "LVL";
    {
        "displayName";
        "Lettischer Lats",
        "displayName-count-one";
        "Lettischer Lats",

```

```

        "displayName-count-other";
        "Lettische Lats",
        "symbol";
        "LVL",
        "symbol-alt-narrow";
        "Ls";
    }
    "LVR";
    {
        "displayName";
        "Lettischer Rubel",
        "displayName-count-one";
        "Lettische Rubel",
        "displayName-count-other";
        "Lettische Rubel",
        "symbol";
        "LVR";
    }
    "LYD";
    {
        "displayName";
        "Libyscher Dinar",
        "displayName-count-one";
        "Libyscher Dinar",
        "displayName-count-other";
        "Libysche Dinar",
        "symbol";
        "LYD";
    }
    "MAD";
    {
        "displayName";
        "Marokkanischer Dirham",
        "displayName-count-one";
        "Marokkanischer Dirham",
        "displayName-count-other";
        "Marokkanische Dirham",
        "symbol";
        "MAD";
    }
    "MAF";
    {
        "displayName";
        "Marokkanischer Franc",
        "displayName-count-one";
        "Marokkanische Franc",
        "displayName-count-other";
        "Marokkanische Franc",
        "symbol";
        "MAF";
    }
    "MCF";
    {
        "displayName";
        "Monegassischer Franc",
        "displayName-count-one";
        "Monegassischer Franc",

```

```

        "displayName-count-other";
        "Monegassische Franc",
        "symbol";
        "MCF";
    }
    "MDC";
    {
        "displayName";
        "Moldau-Cupon",
        "displayName-count-one";
        "Moldau-Cupon",
        "displayName-count-other";
        "Moldau-Cupon",
        "symbol";
        "MDC";
    }
    "MDL";
    {
        "displayName";
        "Moldau-Leu",
        "displayName-count-one";
        "Moldau-Leu",
        "displayName-count-other";
        "Moldau-Leu",
        "symbol";
        "MDL";
    }
    "MGA";
    {
        "displayName";
        "Madagaskar-Ariary",
        "displayName-count-one";
        "Madagaskar-Ariary",
        "displayName-count-other";
        "Madagaskar-Ariary",
        "symbol";
        "MGA",
        "symbol-alt-narrow";
        "Ar";
    }
    "MGF";
    {
        "displayName";
        "Madagaskar-Franc",
        "displayName-count-one";
        "Madagaskar-Franc",
        "displayName-count-other";
        "Madagaskar-Franc",
        "symbol";
        "MGF";
    }
    "MKD";
    {
        "displayName";
        "Mazedonischer Denar",
        "displayName-count-one";
        "Mazedonischer Denar",

```

```

        "displayName-count-other";
        "Mazedonische Denari",
        "symbol";
        "MKD";
    }
    "MKN";
    {
        "displayName";
        "Mazedonischer Denar (1992-1993)",
        "displayName-count-one";
        "Mazedonischer Denar (1992-1993)",
        "displayName-count-other";
        "Mazedonische Denar (1992-1993)",
        "symbol";
        "MKN";
    }
    "MLF";
    {
        "displayName";
        "Malischer Franc",
        "displayName-count-one";
        "Malische Franc",
        "displayName-count-other";
        "Malische Franc",
        "symbol";
        "MLF";
    }
    "MMK";
    {
        "displayName";
        "Myanmarischer Kyat",
        "displayName-count-one";
        "Myanmarischer Kyat",
        "displayName-count-other";
        "Myanmarische Kyat",
        "symbol";
        "MMK",
        "symbol-alt-narrow";
        "K";
    }
    "MNT";
    {
        "displayName";
        "Mongolischer Tögrög",
        "displayName-count-one";
        "Mongolischer Tögrög",
        "displayName-count-other";
        "Mongolische Tögrög",
        "symbol";
        "MNT",
        "symbol-alt-narrow";
        "₮";
    }
    "MOP";
    {
        "displayName";
        "Macao-Pataca",

```

```

        "displayName-count-one";
        "Macao-Pataca",
        "displayName-count-other";
        "Macao-Pataca",
        "symbol";
        "MOP";
    }
    "MRO";
    {
        "displayName";
        "Mauretanischer Ouguiya",
        "displayName-count-one";
        "Mauretanischer Ouguiya",
        "displayName-count-other";
        "Mauretansiche Ouguiya",
        "symbol";
        "MRO";
    }
    "MTL";
    {
        "displayName";
        "Maltesische Lira",
        "displayName-count-one";
        "Maltesische Lira",
        "displayName-count-other";
        "Maltesische Lira",
        "symbol";
        "MTL";
    }
    "MTP";
    {
        "displayName";
        "Maltesisches Pfund",
        "displayName-count-one";
        "Maltesische Pfund",
        "displayName-count-other";
        "Maltesische Pfund",
        "symbol";
        "MTP";
    }
    "MUR";
    {
        "displayName";
        "Mauritius-Rupie",
        "displayName-count-one";
        "Mauritius-Rupie",
        "displayName-count-other";
        "Mauritius-Rupien",
        "symbol";
        "MUR",
        "symbol-alt-narrow";
        "Rs";
    }
    "MVP";
    {
        "displayName";
        "Malediven-Rupie (alt)",

```



```

        "displayName-count-one";
        "Malediven-Rupie (alt)",
        "displayName-count-other";
        "Malediven-Rupien (alt)";
    }
    "MVR";
    {
        "displayName";
        "Malediven-Rufiyaa",
        "displayName-count-one";
        "Malediven-Rufiyaa",
        "displayName-count-other";
        "Malediven-Rupien",
        "symbol";
        "MVR";
    }
    "MWK";
    {
        "displayName";
        "Malawi-Kwacha",
        "displayName-count-one";
        "Malawi-Kwacha",
        "displayName-count-other";
        "Malawi-Kwacha",
        "symbol";
        "MWK";
    }
    "MXN";
    {
        "displayName";
        "Mexikanischer Peso",
        "displayName-count-one";
        "Mexikanischer Peso",
        "displayName-count-other";
        "Mexikanische Pesos",
        "symbol";
        "MX$",
        "symbol-alt-narrow";
        "$";
    }
    "MXP";
    {
        "displayName";
        "Mexikanischer Silber-Peso (1861-1992)",
        "displayName-count-one";
        "Mexikanische Silber-Peso (1861-1992)",
        "displayName-count-other";
        "Mexikanische Silber-Pesos (1861-1992)",
        "symbol";
        "MXP";
    }
    "MXV";
    {
        "displayName";
        "Mexicanischer Unidad de Inversion (UDI)",
        "displayName-count-one";
        "Mexicanischer Unidad de Inversion (UDI)",

```

```

        "displayName-count-other";
        "Mexikanische Unidad de Inversion (UDI)",
        "symbol";
        "MXV";
    }
    "MYR";
    {
        "displayName";
        "Malaysischer Ringgit",
        "displayName-count-one";
        "Malaysischer Ringgit",
        "displayName-count-other";
        "Malaysische Ringgit",
        "symbol";
        "MYR",
        "symbol-alt-narrow";
        "RM";
    }
    "MZE";
    {
        "displayName";
        "Mosambikanischer Escudo",
        "displayName-count-one";
        "Mozambikanische Escudo",
        "displayName-count-other";
        "Mozambikanische Escudo",
        "symbol";
        "MZE";
    }
    "MZM";
    {
        "displayName";
        "Mosambikanischer Metical (1980-2006)",
        "displayName-count-one";
        "Mosambikanischer Metical (1980-2006)",
        "displayName-count-other";
        "Mosambikanische Meticais (1980-2006)",
        "symbol";
        "MZM";
    }
    "MZN";
    {
        "displayName";
        "Mosambikanischer Metical",
        "displayName-count-one";
        "Mosambikanischer Metical",
        "displayName-count-other";
        "Mosambikanische Meticais",
        "symbol";
        "MZN";
    }
    "NAD";
    {
        "displayName";
        "Namibia-Dollar",
        "displayName-count-one";
        "Namibia-Dollar",

```

```

        "displayName-count-other";
        "Namibia-Dollar",
        "symbol";
        "NAD",
        "symbol-alt-narrow";
        "$";
    }
    "NGN";
    {
        "displayName";
        "Nigerianischer Naira",
        "displayName-count-one";
        "Nigerianischer Naira",
        "displayName-count-other";
        "Nigerianische Naira",
        "symbol";
        "NGN",
        "symbol-alt-narrow";
        "₦";
    }
    "NIC";
    {
        "displayName";
        "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-one";
        "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-other";
        "Nicaraguanische Córdoba (1988-1991)",
        "symbol";
        "NIC";
    }
    "NIO";
    {
        "displayName";
        "Nicaragua-Córdoba",
        "displayName-count-one";
        "Nicaragua-Córdoba",
        "displayName-count-other";
        "Nicaragua-Córdobas",
        "symbol";
        "NIO",
        "symbol-alt-narrow";
        "C$";
    }
    "NLG";
    {
        "displayName";
        "Niederländischer Gulden",
        "displayName-count-one";
        "Niederländischer Gulden",
        "displayName-count-other";
        "Niederländische Gulden",
        "symbol";
        "NLG";
    }
    "NOK";
    {

```

```

        "displayName";
        "Norwegische Krone",
        "displayName-count-one";
        "Norwegische Krone",
        "displayName-count-other";
        "Norwegische Kronen",
        "symbol";
        "NOK",
        "symbol-alt-narrow";
        "kr";
    }
    "NPR";
    {
        "displayName";
        "Nepalesische Rupie",
        "displayName-count-one";
        "Nepalesische Rupie",
        "displayName-count-other";
        "Nepalesische Rupien",
        "symbol";
        "NPR",
        "symbol-alt-narrow";
        "Rs";
    }
    "NZD";
    {
        "displayName";
        "Neuseeland-Dollar",
        "displayName-count-one";
        "Neuseeland-Dollar",
        "displayName-count-other";
        "Neuseeland-Dollar",
        "symbol";
        "NZ$",
        "symbol-alt-narrow";
        "$";
    }
    "OMR";
    {
        "displayName";
        "Omanischer Rial",
        "displayName-count-one";
        "Omanischer Rial",
        "displayName-count-other";
        "Omanische Rials",
        "symbol";
        "OMR";
    }
    "PAB";
    {
        "displayName";
        "Panamaischer Balboa",
        "displayName-count-one";
        "Panamaischer Balboa",
        "displayName-count-other";
        "Panamaische Balboas",
        "symbol";
    }

```

```

        "PAB";
    }
    "PEI";
    {
        "displayName";
        "Peruanischer Inti",
        "displayName-count-one";
        "Peruanische Inti",
        "displayName-count-other";
        "Peruanische Inti",
        "symbol";
        "PEI";
    }
    "PEN";
    {
        "displayName";
        "Peruanischer Sol",
        "displayName-count-one";
        "Peruanischer Sol",
        "displayName-count-other";
        "Peruanische Sol",
        "symbol";
        "PEN";
    }
    "PES";
    {
        "displayName";
        "Peruanischer Sol (1863-1965)",
        "displayName-count-one";
        "Peruanischer Sol (1863-1965)",
        "displayName-count-other";
        "Peruanische Sol (1863-1965)",
        "symbol";
        "PES";
    }
    "PGK";
    {
        "displayName";
        "Papua-Neuguineischer Kina",
        "displayName-count-one";
        "Papua-Neuguineischer Kina",
        "displayName-count-other";
        "Papua-Neuguineische Kina",
        "symbol";
        "PGK";
    }
    "PHP";
    {
        "displayName";
        "Philippinischer Peso",
        "displayName-count-one";
        "Philippinischer Peso",
        "displayName-count-other";
        "Philippinische Pesos",
        "symbol";
        "PHP",
        "symbol-alt-narrow";
    }

```

```

        "₨";
    }
    "PKR";
    {
        "displayName";
        "Pakistanische Rupie",
        "displayName-count-one";
        "Pakistanische Rupie",
        "displayName-count-other";
        "Pakistanische Rupien",
        "symbol";
        "PKR",
        "symbol-alt-narrow";
        "Rs";
    }
    "PLN";
    {
        "displayName";
        "Polnischer Złoty",
        "displayName-count-one";
        "Polnischer Złoty",
        "displayName-count-other";
        "Polnische Złoty",
        "symbol";
        "PLN",
        "symbol-alt-narrow";
        "zł";
    }
    "PLZ";
    {
        "displayName";
        "Polnischer Zloty (1950-1995)",
        "displayName-count-one";
        "Polnischer Zloty (1950-1995)",
        "displayName-count-other";
        "Polnische Zloty (1950-1995)",
        "symbol";
        "PLZ";
    }
    "PTE";
    {
        "displayName";
        "Portugiesischer Escudo",
        "displayName-count-one";
        "Portugiesische Escudo",
        "displayName-count-other";
        "Portugiesische Escudo",
        "symbol";
        "PTE";
    }
    "PYG";
    {
        "displayName";
        "Paraguayischer Guaraní",
        "displayName-count-one";
        "Paraguayischer Guaraní",
        "displayName-count-other";
    }

```

```

        "Paraguayische Guaraníes",
        "symbol";
    "PYG",
        "symbol-alt-narrow";
    "₲";
}
"QAR";
{
    "displayName";
    "Katar-Riyal",
        "displayName-count-one";
    "Katar-Riyal",
        "displayName-count-other";
    "Katar-Riyal",
        "symbol";
    "QAR";
}
"RHD";
{
    "displayName";
    "Rhodesischer Dollar",
        "displayName-count-one";
    "Rhodesische Dollar",
        "displayName-count-other";
    "Rhodesische Dollar",
        "symbol";
    "RHD";
}
"ROL";
{
    "displayName";
    "Rumänischer Leu (1952-2006)",
        "displayName-count-one";
    "Rumänischer Leu (1952-2006)",
        "displayName-count-other";
    "Rumänische Leu (1952-2006)",
        "symbol";
    "ROL";
}
"RON";
{
    "displayName";
    "Rumänischer Leu",
        "displayName-count-one";
    "Rumänischer Leu",
        "displayName-count-other";
    "Rumänische Leu",
        "symbol";
    "RON",
        "symbol-alt-narrow";
    "L";
}
"RSD";
{
    "displayName";
    "Serbischer Dinar",
        "displayName-count-one";

```

```

        "Serbischer Dinar",
        "displayName-count-other";
    "Serbische Dinaren",
        "symbol";
    "RSD";
}
"RUB";
{
    "displayName";
    "Russischer Rubel",
        "displayName-count-one";
    "Russischer Rubel",
        "displayName-count-other";
    "Russische Rubel",
        "symbol";
    "RUB",
        "symbol-alt-narrow";
    "₽";
}
"RUR";
{
    "displayName";
    "Russischer Rubel (1991-1998)",
        "displayName-count-one";
    "Russischer Rubel (1991-1998)",
        "displayName-count-other";
    "Russische Rubel (1991-1998)",
        "symbol";
    "RUR",
        "symbol-alt-narrow";
    "p.";
}
"RWF";
{
    "displayName";
    "Ruanda-Franc",
        "displayName-count-one";
    "Ruanda-Franc",
        "displayName-count-other";
    "Ruanda-Francs",
        "symbol";
    "RWF",
        "symbol-alt-narrow";
    "F.Rw";
}
"SAR";
{
    "displayName";
    "Saudi-Rial",
        "displayName-count-one";
    "Saudi-Rial",
        "displayName-count-other";
    "Saudi-Rial",
        "symbol";
    "SAR";
}
"SBD";

```



```

    {
      "displayName";
      "Salomonen-Dollar",
      "displayName-count-one";
      "Salomonen-Dollar",
      "displayName-count-other";
      "Salomonen-Dollar",
      "symbol";
      "SBD",
      "symbol-alt-narrow";
      "$";
    }
    "SCR";
    {
      "displayName";
      "Seychellen-Rupie",
      "displayName-count-one";
      "Seychellen-Rupie",
      "displayName-count-other";
      "Seychellen-Rupien",
      "symbol";
      "SCR";
    }
    "SDD";
    {
      "displayName";
      "Sudanesischer Dinar (1992-2007)",
      "displayName-count-one";
      "Sudanesischer Dinar (1992-2007)",
      "displayName-count-other";
      "Sudanesische Dinar (1992-2007)",
      "symbol";
      "SDD";
    }
    "SDG";
    {
      "displayName";
      "Sudanesisches Pfund",
      "displayName-count-one";
      "Sudanesisches Pfund",
      "displayName-count-other";
      "Sudanesische Pfund",
      "symbol";
      "SDG";
    }
    "SDP";
    {
      "displayName";
      "Sudanesisches Pfund (1957-1998)",
      "displayName-count-one";
      "Sudanesisches Pfund (1957-1998)",
      "displayName-count-other";
      "Sudanesische Pfund (1957-1998)",
      "symbol";
      "SDP";
    }
    "SEK";

```

```

    {
      "displayName";
      "Schwedische Krone",
        "displayName-count-one";
      "Schwedische Krone",
        "displayName-count-other";
      "Schwedische Kronen",
        "symbol";
      "SEK",
        "symbol-alt-narrow";
      "kr";
    }
    "SGD";
    {
      "displayName";
      "Singapur-Dollar",
        "displayName-count-one";
      "Singapur-Dollar",
        "displayName-count-other";
      "Singapur-Dollar",
        "symbol";
      "SGD",
        "symbol-alt-narrow";
      "$";
    }
    "SHP";
    {
      "displayName";
      "St. Helena-Pfund",
        "displayName-count-one";
      "St. Helena-Pfund",
        "displayName-count-other";
      "St. Helena-Pfund",
        "symbol";
      "SHP",
        "symbol-alt-narrow";
      "£";
    }
    "SIT";
    {
      "displayName";
      "Slowenischer Tolar",
        "displayName-count-one";
      "Slowenischer Tolar",
        "displayName-count-other";
      "Slowenische Tolar",
        "symbol";
      "SIT";
    }
    "SKK";
    {
      "displayName";
      "Slowakische Krone",
        "displayName-count-one";
      "Slowakische Kronen",
        "displayName-count-other";
      "Slowakische Kronen",

```

```

        "symbol";
        "SKK";
    }
    "SLL";
    {
        "displayName";
        "Sierra-leonischer Leone",
        "displayName-count-one";
        "Sierra-leonischer Leone",
        "displayName-count-other";
        "Sierra-leonische Leones",
        "symbol";
        "SLL";
    }
    "SOS";
    {
        "displayName";
        "Somalia-Schilling",
        "displayName-count-one";
        "Somalia-Schilling",
        "displayName-count-other";
        "Somalia-Schilling",
        "symbol";
        "SOS";
    }
    "SRD";
    {
        "displayName";
        "Suriname-Dollar",
        "displayName-count-one";
        "Suriname-Dollar",
        "displayName-count-other";
        "Suriname-Dollar",
        "symbol";
        "SRD",
        "symbol-alt-narrow";
        "$";
    }
    "SRG";
    {
        "displayName";
        "Suriname Gulden",
        "displayName-count-one";
        "Suriname-Gulden",
        "displayName-count-other";
        "Suriname-Gulden",
        "symbol";
        "SRG";
    }
    "SSP";
    {
        "displayName";
        "Südsudanesisches Pfund",
        "displayName-count-one";
        "Südsudanesisches Pfund",
        "displayName-count-other";
        "Südsudanesische Pfund",

```

```

        "symbol";
        "SSP",
        "symbol-alt-narrow";
        "£";
    }
    "STD";
    {
        "displayName";
        "São-toméischer Dobra",
        "displayName-count-one";
        "São-toméischer Dobra",
        "displayName-count-other";
        "São-toméische Dobra",
        "symbol";
        "STD",
        "symbol-alt-narrow";
        "Db";
    }
    "SUR";
    {
        "displayName";
        "Sowjetischer Rubel",
        "displayName-count-one";
        "Sowjetische Rubel",
        "displayName-count-other";
        "Sowjetische Rubel",
        "symbol";
        "SUR";
    }
    "SVC";
    {
        "displayName";
        "El Salvador Colon",
        "displayName-count-one";
        "El Salvador-Colon",
        "displayName-count-other";
        "El Salvador-Colon",
        "symbol";
        "SVC";
    }
    "SYP";
    {
        "displayName";
        "Syrisches Pfund",
        "displayName-count-one";
        "Syrisches Pfund",
        "displayName-count-other";
        "Syrische Pfund",
        "symbol";
        "SYP",
        "symbol-alt-narrow";
        "SYP";
    }
    "SZL";
    {
        "displayName";
        "Swasiländischer Lilangeni",

```

```

        "displayName-count-one";
        "Swasiländischer Lilangeni",
        "displayName-count-other";
        "Swasiländische Emalangeni",
        "symbol";
        "SZL";
    }
    "THB";
    {
        "displayName";
        "Thailändischer Baht",
        "displayName-count-one";
        "Thailändischer Baht",
        "displayName-count-other";
        "Thailändische Baht",
        "symbol";
        "฿",
        "symbol-alt-narrow";
        "฿";
    }
    "TJR";
    {
        "displayName";
        "Tadschikistan Rubel",
        "displayName-count-one";
        "Tadschikistan-Rubel",
        "displayName-count-other";
        "Tadschikistan-Rubel",
        "symbol";
        "TJR";
    }
    "TJS";
    {
        "displayName";
        "Tadschikistan-Somoni",
        "displayName-count-one";
        "Tadschikistan-Somoni",
        "displayName-count-other";
        "Tadschikistan-Somoni",
        "symbol";
        "TJS";
    }
    "TMM";
    {
        "displayName";
        "Turkmenistan-Manat (1993-2009)",
        "displayName-count-one";
        "Turkmenistan-Manat (1993-2009)",
        "displayName-count-other";
        "Turkmenistan-Manat (1993-2009)",
        "symbol";
        "TMM";
    }
    "TMT";
    {
        "displayName";
        "Turkmenistan-Manat",

```

```

        "displayName-count-one";
        "Turkmenistan-Manat",
        "displayName-count-other";
        "Turkmenistan-Manat",
        "symbol";
        "TMT";
    }
    "TND";
    {
        "displayName";
        "Tunesischer Dinar",
        "displayName-count-one";
        "Tunesischer Dinar",
        "displayName-count-other";
        "Tunesische Dinar",
        "symbol";
        "TND";
    }
    "TOP";
    {
        "displayName";
        "Tongaischer Pa'anga",
        "displayName-count-one";
        "Tongaischer Pa'anga",
        "displayName-count-other";
        "Tongaische Pa'anga",
        "symbol";
        "TOP",
        "symbol-alt-narrow";
        "T$";
    }
    "TPE";
    {
        "displayName";
        "Timor-Escudo",
        "displayName-count-one";
        "Timor-Escudo",
        "displayName-count-other";
        "Timor-Escudo",
        "symbol";
        "TPE";
    }
    "TRL";
    {
        "displayName";
        "Türkische Lira (1922-2005)",
        "displayName-count-one";
        "Türkische Lira (1922-2005)",
        "displayName-count-other";
        "Türkische Lira (1922-2005)",
        "symbol";
        "TRL";
    }
    "TRY";
    {
        "displayName";
        "Türkische Lira",

```

```

        "displayName-count-one";
        "Türkische Lira",
        "displayName-count-other";
        "Türkische Lira",
        "symbol";
        "TRY",
        "symbol-alt-narrow";
        "₺",
        "symbol-alt-variant";
        "TL";
    }
    "TTD";
    {
        "displayName";
        "Trinidad und Tobago-Dollar",
        "displayName-count-one";
        "Trinidad und Tobago-Dollar",
        "displayName-count-other";
        "Trinidad und Tobago-Dollar",
        "symbol";
        "TTD",
        "symbol-alt-narrow";
        "$";
    }
    "TWD";
    {
        "displayName";
        "Neuer Taiwan-Dollar",
        "displayName-count-one";
        "Neuer Taiwan-Dollar",
        "displayName-count-other";
        "Neue Taiwan-Dollar",
        "symbol";
        "NT$",
        "symbol-alt-narrow";
        "NT$";
    }
    "TZS";
    {
        "displayName";
        "Tansania-Schilling",
        "displayName-count-one";
        "Tansania-Schilling",
        "displayName-count-other";
        "Tansania-Schilling",
        "symbol";
        "TZS";
    }
    "UAH";
    {
        "displayName";
        "Ukrainische Hrywnja",
        "displayName-count-one";
        "Ukrainische Hrywnja",
        "displayName-count-other";
        "Ukrainische Hrywen",
        "symbol";
    }

```

```

        "UAH",
        "symbol-alt-narrow";
        "₴";
    }
    "UAK";
    {
        "displayName";
        "Ukrainischer Karbovanetz",
        "displayName-count-one";
        "Ukrainische Karbovanetz",
        "displayName-count-other";
        "Ukrainische Karbovanetz",
        "symbol";
        "UAK";
    }
    "UGS";
    {
        "displayName";
        "Uganda-Schilling (1966-1987)",
        "displayName-count-one";
        "Uganda-Schilling (1966-1987)",
        "displayName-count-other";
        "Uganda-Schilling (1966-1987)",
        "symbol";
        "UGS";
    }
    "UGX";
    {
        "displayName";
        "Uganda-Schilling",
        "displayName-count-one";
        "Uganda-Schilling",
        "displayName-count-other";
        "Uganda-Schilling",
        "symbol";
        "UGX";
    }
    "USD";
    {
        "displayName";
        "US-Dollar",
        "displayName-count-one";
        "US-Dollar",
        "displayName-count-other";
        "US-Dollar",
        "symbol";
        "$",
        "symbol-alt-narrow";
        "$";
    }
    "USN";
    {
        "displayName";
        "US Dollar (Nächster Tag)",
        "displayName-count-one";
        "US-Dollar (Nächster Tag)",
        "displayName-count-other";
    }

```



```

        "US-Dollar (Nächster Tag)",
        "symbol";
        "USN";
    }
    "USS";
    {
        "displayName";
        "US Dollar (Gleicher Tag)",
        "displayName-count-one";
        "US-Dollar (Gleicher Tag)",
        "displayName-count-other";
        "US-Dollar (Gleicher Tag)",
        "symbol";
        "USS";
    }
    "UYI";
    {
        "displayName";
        "Uruguayischer Peso (Indexierte
Rechnungseinheiten)",
        "displayName-count-one";
        "Uruguayischer Peso (Indexierte
Rechnungseinheiten)",
        "displayName-count-other";
        "Uruguayische Pesos (Indexierte
Rechnungseinheiten)",
        "symbol";
        "UYI";
    }
    "UYP";
    {
        "displayName";
        "Uruguayischer Peso (1975-1993)",
        "displayName-count-one";
        "Uruguayischer Peso (1975-1993)",
        "displayName-count-other";
        "Uruguayische Pesos (1975-1993)",
        "symbol";
        "UYP";
    }
    "UYU";
    {
        "displayName";
        "Uruguayischer Peso",
        "displayName-count-one";
        "Uruguayischer Peso",
        "displayName-count-other";
        "Uruguayische Pesos",
        "symbol";
        "UYU",
        "symbol-alt-narrow";
        "$";
    }
    "UZS";
    {
        "displayName";
        "Usbekistan-Sum",

```

```

        "displayName-count-one";
        "Usbekistan-Sum",
        "displayName-count-other";
        "Usbekistan-Sum",
        "symbol";
        "UZS";
    }
    "VEB";
    {
        "displayName";
        "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-one";
        "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-other";
        "Venezolanische Bolívares (1871-2008)",
        "symbol";
        "VEB";
    }
    "VEF";
    {
        "displayName";
        "Venezolanischer Bolívar",
        "displayName-count-one";
        "Venezolanischer Bolívar",
        "displayName-count-other";
        "Venezolanische Bolívares",
        "symbol";
        "VEF",
        "symbol-alt-narrow";
        "Bs";
    }
    "VND";
    {
        "displayName";
        "Vietnamesischer Dong",
        "displayName-count-one";
        "Vietnamesischer Dong",
        "displayName-count-other";
        "Vietnamesische Dong",
        "symbol";
        "₫",
        "symbol-alt-narrow";
        "₫";
    }
    "VNN";
    {
        "displayName";
        "Vietnamesischer Dong (1978-1985)",
        "displayName-count-one";
        "Vietnamesischer Dong (1978-1985)",
        "displayName-count-other";
        "Vietnamesische Dong (1978-1985)",
        "symbol";
        "VNN";
    }
    "VUV";
    {

```

```

        "displayName";
        "Vanuatu-Vatu",
        "displayName-count-one";
        "Vanuatu-Vatu",
        "displayName-count-other";
        "Vanuatu-Vatu",
        "symbol";
        "VUV";
    }
    "WST";
    {
        "displayName";
        "Samoanischer Tala",
        "displayName-count-one";
        "Samoanischer Tala",
        "displayName-count-other";
        "Samoanische Tala",
        "symbol";
        "WST";
    }
    "XAF";
    {
        "displayName";
        "CFA-Franc (BEAC) ",
        "displayName-count-one";
        "CFA-Franc (BEAC) ",
        "displayName-count-other";
        "CFA-Franc (BEAC) ",
        "symbol";
        "FCFA";
    }
    "XAG";
    {
        "displayName";
        "Unze Silber",
        "displayName-count-one";
        "Unze Silber",
        "displayName-count-other";
        "Unzen Silber",
        "symbol";
        "XAG";
    }
    "XAU";
    {
        "displayName";
        "Unze Gold",
        "displayName-count-one";
        "Unze Gold",
        "displayName-count-other";
        "Unzen Gold",
        "symbol";
        "XAU";
    }
    "XBA";
    {
        "displayName";
        "Europäische Rechnungseinheit",

```

```

        "displayName-count-one";
        "Europäische Rechnungseinheiten",
        "displayName-count-other";
        "Europäische Rechnungseinheiten",
        "symbol";
        "XBA";
    }
    "XBB";
    {
        "displayName";
        "Europäische Währungseinheit (XBB)",
        "displayName-count-one";
        "Europäische Währungseinheiten (XBB)",
        "displayName-count-other";
        "Europäische Währungseinheiten (XBB)",
        "symbol";
        "XBB";
    }
    "XBC";
    {
        "displayName";
        "Europäische Rechnungseinheit (XBC)",
        "displayName-count-one";
        "Europäische Rechnungseinheiten (XBC)",
        "displayName-count-other";
        "Europäische Rechnungseinheiten (XBC)",
        "symbol";
        "XBC";
    }
    "XBD";
    {
        "displayName";
        "Europäische Rechnungseinheit (XBD)",
        "displayName-count-one";
        "Europäische Rechnungseinheiten (XBD)",
        "displayName-count-other";
        "Europäische Rechnungseinheiten (XBD)",
        "symbol";
        "XBD";
    }
    "XCD";
    {
        "displayName";
        "Ostkaribischer Dollar",
        "displayName-count-one";
        "Ostkaribischer Dollar",
        "displayName-count-other";
        "Ostkaribische Dollar",
        "symbol";
        "EC$",
        "symbol-alt-narrow";
        "$";
    }
    "XDR";
    {
        "displayName";
        "Sonderziehungsrechte",

```

```

        "displayName-count-one";
        "Sonderziehungsrechte",
        "displayName-count-other";
        "Sonderziehungsrechte",
        "symbol";
        "XDR";
    }
    "XEU";
    {
        "displayName";
        "Europäische Währungseinheit (XEU)",
        "displayName-count-one";
        "Europäische Währungseinheiten (XEU)",
        "displayName-count-other";
        "Europäische Währungseinheiten (XEU)",
        "symbol";
        "XEU";
    }
    "XFO";
    {
        "displayName";
        "Französischer Gold-Franc",
        "displayName-count-one";
        "Französische Gold-Franc",
        "displayName-count-other";
        "Französische Gold-Franc",
        "symbol";
        "XFO";
    }
    "XFU";
    {
        "displayName";
        "Französischer UIC-Franc",
        "displayName-count-one";
        "Französische UIC-Franc",
        "displayName-count-other";
        "Französische UIC-Franc",
        "symbol";
        "XFU";
    }
    "XOF";
    {
        "displayName";
        "CFA-Franc (BCEAO)",
        "displayName-count-one";
        "CFA-Franc (BCEAO)",
        "displayName-count-other";
        "CFA-Francs (BCEAO)",
        "symbol";
        "CFA";
    }
    "XPD";
    {
        "displayName";
        "Unze Palladium",
        "displayName-count-one";
        "Unze Palladium",

```

```

        "displayName-count-other";
        "Unzen Palladium",
        "symbol";
        "XPD";
    }
    "XPF";
    {
        "displayName";
        "CFP-Franc",
        "displayName-count-one";
        "CFP-Franc",
        "displayName-count-other";
        "CFP-Franc",
        "symbol";
        "CFPF";
    }
    "XPT";
    {
        "displayName";
        "Unze Platin",
        "displayName-count-one";
        "Unze Platin",
        "displayName-count-other";
        "Unzen Platin",
        "symbol";
        "XPT";
    }
    "XRE";
    {
        "displayName";
        "RINET Funds",
        "displayName-count-one";
        "RINET Funds",
        "displayName-count-other";
        "RINET Funds",
        "symbol";
        "XRE";
    }
    "XSU";
    {
        "displayName";
        "SUCRE",
        "displayName-count-one";
        "SUCRE",
        "displayName-count-other";
        "SUCRE",
        "symbol";
        "XSU";
    }
    "XTS";
    {
        "displayName";
        "Testwährung",
        "displayName-count-one";
        "Testwährung",
        "displayName-count-other";
        "Testwährung",

```

```

        "symbol";
        "XTS";
    }
    "XUA";
    {
        "displayName";
        "Rechnungseinheit der AfEB",
        "displayName-count-one";
        "Rechnungseinheit der AfEB",
        "displayName-count-other";
        "Rechnungseinheiten der AfEB",
        "symbol";
        "XUA";
    }
    "XXX";
    {
        "displayName";
        "Unbekannte Währung",
        "displayName-count-one";
        "(unbekannte Währung)",
        "displayName-count-other";
        "(unbekannte Währung)",
        "symbol";
        "XXX";
    }
    "YDD";
    {
        "displayName";
        "Jemen-Dinar",
        "displayName-count-one";
        "Jemen-Dinar",
        "displayName-count-other";
        "Jemen-Dinar",
        "symbol";
        "YDD";
    }
    "YER";
    {
        "displayName";
        "Jemen-Rial",
        "displayName-count-one";
        "Jemen-Rial",
        "displayName-count-other";
        "Jemen-Rial",
        "symbol";
        "YER";
    }
    "YUD";
    {
        "displayName";
        "Jugoslawischer Dinar (1966-1990)",
        "displayName-count-one";
        "Jugoslawischer Dinar (1966-1990)",
        "displayName-count-other";
        "Jugoslawische Dinar (1966-1990)",
        "symbol";
        "YUD";
    }

```

```

    }
    "YUM";
    {
        "displayName";
        "Jugoslawischer Neuer Dinar (1994-2002)",
        "displayName-count-one";
        "Jugoslawischer Neuer Dinar (1994-2002)",
        "displayName-count-other";
        "Jugoslawische Neue Dinar (1994-2002)",
        "symbol";
        "YUM";
    }
    "YUN";
    {
        "displayName";
        "Jugoslawischer Dinar (konvertibel)",
        "displayName-count-one";
        "Jugoslawische Dinar (konvertibel)",
        "displayName-count-other";
        "Jugoslawische Dinar (konvertibel)",
        "symbol";
        "YUN";
    }
    "YUR";
    {
        "displayName";
        "Jugoslawischer reformierter Dinar (1992-1993)",
        "displayName-count-one";
        "Jugoslawischer reformierter Dinar (1992-1993)",
        "displayName-count-other";
        "Jugoslawische reformierte Dinar (1992-1993)",
        "symbol";
        "YUR";
    }
    "ZAL";
    {
        "displayName";
        "Südafrikanischer Rand (Finanz)",
        "displayName-count-one";
        "Südafrikanischer Rand (Finanz)",
        "displayName-count-other";
        "Südafrikanischer Rand (Finanz)",
        "symbol";
        "ZAL";
    }
    "ZAR";
    {
        "displayName";
        "Südafrikanischer Rand",
        "displayName-count-one";
        "Südafrikanischer Rand",
        "displayName-count-other";
        "Südafrikanische Rand",
        "symbol";
        "ZAR",
        "symbol-alt-narrow";
        "R";
    }

```



```

    }
    "ZMK";
    {
        "displayName";
        "Kwacha (1968-2012)",
        "displayName-count-one";
        "Kwacha (1968-2012)",
        "displayName-count-other";
        "Kwacha (1968-2012)",
        "symbol";
        "ZMK";
    }
    "ZMW";
    {
        "displayName";
        "Kwacha",
        "displayName-count-one";
        "Kwacha",
        "displayName-count-other";
        "Kwacha",
        "symbol";
        "ZMW",
        "symbol-alt-narrow";
        "K";
    }
    "ZRN";
    {
        "displayName";
        "Zaire-Neuer Zaïre (1993-1998)",
        "displayName-count-one";
        "Zaire-Neuer Zaïre (1993-1998)",
        "displayName-count-other";
        "Zaire-Neue Zaïre (1993-1998)",
        "symbol";
        "ZRN";
    }
    "ZRZ";
    {
        "displayName";
        "Zaire-Zaïre (1971-1993)",
        "displayName-count-one";
        "Zaire-Zaïre (1971-1993)",
        "displayName-count-other";
        "Zaire-Zaïre (1971-1993)",
        "symbol";
        "ZRZ";
    }
    "ZWD";
    {
        "displayName";
        "Simbabwe-Dollar (1980-2008)",
        "displayName-count-one";
        "Simbabwe-Dollar (1980-2008)",
        "displayName-count-other";
        "Simbabwe-Dollar (1980-2008)",
        "symbol";
        "ZWD";
    }

```

```
    }  
    "ZWL";  
    {  
        "displayName";  
        "Simbabwe-Dollar (2009)",  
            "displayName-count-one";  
        "Simbabwe-Dollar (2009)",  
            "displayName-count-other";  
        "Simbabwe-Dollar (2009)",  
            "symbol";  
        "ZWL";  
    }  
    "ZWR";  
    {  
        "displayName";  
        "Simbabwe-Dollar (2008)",  
            "displayName-count-one";  
        "Simbabwe-Dollar (2008)",  
            "displayName-count-other";  
        "Simbabwe-Dollar (2008)",  
            "symbol";  
        "ZWR";  
    }  
}  
  
}  
  
}
```

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
//import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import { loadCldr, L10n } from '@syncfusion/ej2-base';
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as timeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames);
L10n.load({
    'de': {
        'calendar': { today: 'heute' }
    }
});
function App() {
    return <CalendarComponent id="calendar" locale='de'/>;
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import * as React from "react";
```

```
import * as ReactDOM from "react-dom";
//import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import { loadCldr, L10n } from '@syncfusion/ej2-base';
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as timeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames);
L10n.load({
  'de': {
    'calendar': { today: 'heute' }
  }
});
function App() {
  return <CalendarComponent id="calendar" locale='de' />
};
ReactDOM.render(<App />, document.getElementById('element'));
```

NUMBERINGSYSTEMS.JSON

```
{
  "supplemental": {
    "version": {
      "_number": "$Revision: 12732 $",
      "_unicodeVersion": "9.0.0",
      "_cldrVersion": "31"
    },
    "numberingSystems": {
      "adlm": {
        "_digits": "ᲠᲡᲢᲣᲤᲥᲦᲧᲨᲩ",
        "_type": "numeric"
      },
      "ahom": {
        "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉",
        "_type": "numeric"
      },
      "arab": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "arabext": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "armn": {
        "_rules": "armenian-upper",
        "_type": "algorithmic"
      },
      "armnlow": {
        "_rules": "armenian-lower",
        "_type": "algorithmic"
      },
      "bali": {
        "_digits": "ᬀᬁᬂᬃᬄᬅᬆᬇᬈᬉ",

```

```

    "_type": "numeric"
  },
  "beng": {
    "_digits": "০১২৩৪৫৬৭৮৯",
    "_type": "numeric"
  },
  "bhks": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "brah": {
    "_digits": "·٠١٢٣٤٥٦٧٨٩",
    "_type": "numeric"
  },
  "cakm": {
    "_digits": "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
    "_type": "numeric"
  },
  "cham": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "cyr1": {
    "_rules": "cyrillic-lower",
    "_type": "algorithmic"
  },
  "deva": {
    "_digits": "०१२३४५६७८९",
    "_type": "numeric"
  },
  "ethi": {
    "_rules": "ethiopic",
    "_type": "algorithmic"
  },
  "fullwide": {
    "_digits": "0 1 2 3 4 5 6 7 8 9",
    "_type": "numeric"
  },
  "geor": {
    "_rules": "georgian",
    "_type": "algorithmic"
  },
  "grek": {
    "_rules": "greek-upper",
    "_type": "algorithmic"
  },
  "greklow": {
    "_rules": "greek-lower",
    "_type": "algorithmic"
  },
  "gujr": {
    "_digits": "૦૧૨૩૪૫૬૭૮૯",
    "_type": "numeric"
  },
  "guru": {

```

```

    "_digits": "၀၇၃၃၈၄၆၅၇၉",
    "_type": "numeric"
  },
  "hanidays": {
    "_rules": "zh/SpelloutRules/spellout-numbering-days",
    "_type": "algorithmic"
  },
  "hanidec": {
    "_digits": "〇一二三四五六七八九",
    "_type": "numeric"
  },
  "hans": {
    "_rules": "zh/SpelloutRules/spellout-cardinal",
    "_type": "algorithmic"
  },
  "hansfin": {
    "_rules": "zh/SpelloutRules/spellout-cardinal-financial",
    "_type": "algorithmic"
  },
  "hant": {
    "_rules": "zh_Hant/SpelloutRules/spellout-cardinal",
    "_type": "algorithmic"
  },
  "hantfin": {
    "_rules": "zh_Hant/SpelloutRules/spellout-cardinal-financial",
    "_type": "algorithmic"
  },
  "hebr": {
    "_rules": "hebrew",
    "_type": "algorithmic"
  },
  "hmng": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "java": {
    "_digits": "၀၈၈၅၅၆၇၉၀၁၁၂၁၃၁၄",
    "_type": "numeric"
  },
  "jpan": {
    "_rules": "ja/SpelloutRules/spellout-cardinal",
    "_type": "algorithmic"
  },
  "jpanfin": {
    "_rules": "ja/SpelloutRules/spellout-cardinal-financial",
    "_type": "algorithmic"
  },
  "kali": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "khmr": {
    "_digits": "០១២៣៤៥៦៧៨៩",
    "_type": "numeric"
  },

```

```

"knnda": {
  "_digits": "೦೧೨೩೪೫೬೭೮೯",
  "_type": "numeric"
},
"lana": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"lanatham": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"laoo": {
  "_digits": "໐໑໒໓໔໕໖໗໘໙",
  "_type": "numeric"
},
"latn": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"lepc": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"limb": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"mathbold": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathdbl": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathmono": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathsanb": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathsans": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mlym": {
  "_digits": "൦൧൨൩൪൫൬൭൮൯",
  "_type": "numeric"
},
"modi": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
}

```

```

    },
    "mong": {
      "_digits": "᠐᠑᠒᠓᠔᠕᠖᠗᠘᠐",
      "_type": "numeric"
    },
    "mroo": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "mtei": {
      "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍᱎᱏ᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱞᱯᱰᱴᱶᱺᱼᱽ᱾᱿",
      "_type": "numeric"
    },
    "mymr": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "mymrshan": {
      "_digits": "0123456789",
      "_type": "numeric"
    },
    "mymrtlng": {
      "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍᱎᱏ᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱞᱯᱰᱴᱶᱺᱼᱽ᱾᱿",
      "_type": "numeric"
    },
    "newa": {
      "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍᱎᱏ᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱞᱯᱰᱴᱶᱺᱼᱽ᱾᱿",
      "_type": "numeric"
    },
    "nkoo": {
      "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍᱎᱏ᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱞᱯᱰᱴᱶᱺᱼᱽ᱾᱿",
      "_type": "numeric"
    },
    "olck": {
      "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍᱎᱏ᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱞᱯᱰᱴᱶᱺᱼᱽ᱾᱿",
      "_type": "numeric"
    },
    "orya": {
      "_digits": "୦୧୨୩୪୫୬୭୮୯",
      "_type": "numeric"
    },
    "osma": {
      "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍᱎᱏ᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱞᱯᱰᱴᱶᱺᱼᱽ᱾᱿",
      "_type": "numeric"
    },
    "roman": {
      "_rules": "roman-upper",
      "_type": "algorithmic"
    },
    "romanlow": {
      "_rules": "roman-lower",
      "_type": "algorithmic"
    },
  },

```

```

"saur": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"shrd": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"sind": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"sinh": {
  "_digits": "ආශ්වාතචූර්ණි",
  "_type": "numeric"
},
"sora": {
  "_digits": "ᱚᱛᱟᱨᱛᱟᱲ",
  "_type": "numeric"
},
"sund": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"takr": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"talv": {
  "_digits": "တၢ်သၢ်လၢ",
  "_type": "numeric"
},
"taml": {
  "_rules": "tamil",
  "_type": "algorithmic"
},
"tamldec": {
  "_digits": "௦௧௨௩௪௫௬௭௮௯",
  "_type": "numeric"
},
"telu": {
  "_digits": "౦౧౨౩౪౫౬౭౮౯",
  "_type": "numeric"
},
"thai": {
  "_digits": "๐๑๒๓๔๕๖๗๘๙",
  "_type": "numeric"
},
"tibb": {
  "_digits": "༠༡༢༣༤༥༦༧༨༩",
  "_type": "numeric"
},
"tirh": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
}

```



```

    },
    "vaii": {
      "_digits": "᠐᠑᠒᠓᠔᠕᠖᠗᠘᠐᠑᠒",
      "_type": "numeric"
    },
    "wara": {
      "_digits": "᠐᠐᠐᠐᠐᠐᠐᠐᠐᠐",
      "_type": "numeric"
    }
  }
}

```

NUMBERINGSYSTEMS.JSX

```

{
  "supplemental";
  {
    "version";
    {
      "_number";
      "$Revision: 12732 $",
      "_unicodeVersion";
      "9.0.0",
      "_cldrVersion";
      "31";
    }
    "numberingSystems";
    {
      "adlm";
      {
        "_digits";
        "ᠠᠳᠯᠮᠠᠳᠯᠮᠠᠳᠯᠮ",
        "_type";
        "numeric";
      }
      "ahom";
      {
        "_digits";
        "ᠠᠬᠣᠮᠠᠬᠣᠮᠠᠬᠣᠮ",
        "_type";
        "numeric";
      }
      "arab";
      {
        "_digits";
        "٠١٢٣٤٥٦٧٨٩",
        "_type";
        "numeric";
      }
      "arabext";
      {
        "_digits";
        "٠١٢٣٤٥٦٧٨٩",
        "_type";
        "numeric";
      }
    }
  }
}

```

```

    }
    "armn";
    {
        "_rules";
        "armenian-upper",
        "_type";
        "algorithmic";
    }
    "armnlow";
    {
        "_rules";
        "armenian-lower",
        "_type";
        "algorithmic";
    }
    "bali";
    {
        "_digits";
        "ᮀᮁᮂᮃᮄᮅᮆᮇᮈᮉ",
        "_type";
        "numeric";
    }
    "beng";
    {
        "_digits";
        "০১২৩৪৫৬৭৮৯",
        "_type";
        "numeric";
    }
    "bhks";
    {
        "_digits";
        "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉",
        "_type";
        "numeric";
    }
    "brah";
    {
        "_digits";
        "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉",
        "_type";
        "numeric";
    }
    "cakm";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "cham";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }

```

```

    }
    "cyr1";
    {
        "_rules";
        "cyrillic-lower",
        "_type";
        "algorithmic";
    }
    "deva";
    {
        "_digits";
        "०१२३४५६७८९",
        "_type";
        "numeric";
    }
    "ethi";
    {
        "_rules";
        "ethiopic",
        "_type";
        "algorithmic";
    }
    "fullwide";
    {
        "_digits";
        "0 1 2 3 4 5 6 7 8 9",
        "_type";
        "numeric";
    }
    "geor";
    {
        "_rules";
        "georgian",
        "_type";
        "algorithmic";
    }
    "grek";
    {
        "_rules";
        "greek-upper",
        "_type";
        "algorithmic";
    }
    "greklow";
    {
        "_rules";
        "greek-lower",
        "_type";
        "algorithmic";
    }
    "gujr";
    {
        "_digits";
        "૦૧૨૩૪૫૬૭૮૯",
        "_type";
        "numeric";
    }

```

```

    }
    "guru";
    {
        "_digits";
        "୦୧୨୩୪୫୬୭୮୯",
        "_type";
        "numeric";
    }
    "hanidays";
    {
        "_rules";
        "zh/SpelloutRules/spellout-numbering-days",
        "_type";
        "algorithmic";
    }
    "hanidec";
    {
        "_digits";
        "〇一二三四五六七八九",
        "_type";
        "numeric";
    }
    "hans";
    {
        "_rules";
        "zh/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "hansfin";
    {
        "_rules";
        "zh/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "hant";
    {
        "_rules";
        "zh_Hant/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "hantfin";
    {
        "_rules";
        "zh_Hant/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "hebr";
    {
        "_rules";
        "hebrew",
        "_type";
        "algorithmic";
    }

```

```

"hmng";
{
  "_digits";
  "□□□□□□□□□□",
  "_type";
  "numeric";
}
"java";
{
  "_digits";
  "௦௧௨௩௪௫௬௭௮௯௦",
  "_type";
  "numeric";
}
"jpan";
{
  "_rules";
  "ja/SpelloutRules/spellout-cardinal",
  "_type";
  "algorithmic";
}
"jpanfin";
{
  "_rules";
  "ja/SpelloutRules/spellout-cardinal-financial",
  "_type";
  "algorithmic";
}
"kali";
{
  "_digits";
  "□□□□□□□□□□",
  "_type";
  "numeric";
}
"khmr";
{
  "_digits";
  "០១២៣៤៥៦៧៨៩",
  "_type";
  "numeric";
}
"knda";
{
  "_digits";
  "೦೧೨೩೪೫೬೭೮೯",
  "_type";
  "numeric";
}
"lana";
{
  "_digits";
  "□□□□□□□□□□",
  "type";

```

```

        "numeric";
    }
    "lanatham";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "laoo";
    {
        "_digits";
        "ອາໄສໄລຍະ",
        "_type";
        "numeric";
    }
    "latn";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "lepc";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "limb";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "mathbold";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathdbl";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathmono";
    {
        "_digits";
        "0123456789",
        "_type";

```

```
"numeric";
}
"mathsanb";
{
    "_digits";
    "0123456789",
    "_type";
    "numeric";
}
"mathsans";
{
    "_digits";
    "0123456789",
    "_type";
    "numeric";
}
"mlym";
{
    "_digits";
    "ഐഹുരൂറുനവുൻ",
    "_type";
    "numeric";
}
"modi";
{
    "_digits";
    "□□□□□□□□",
    "_type";
    "numeric";
}
"mong";
{
    "_digits";
    "᠎ᠠᠨᠮᠤᠩᠭᠡ",
    "_type";
    "numeric";
}
"mroo";
{
    "_digits";
    "□□□□□□□□",
    "_type";
    "numeric";
}
"mtei";
{
    "_digits";
    "ᱟᱱᱟᱛᱚᱸᱰᱽᱯᱷᱚᱴᱚ",
    "_type";
    "numeric";
}
"mymr";
{
    "_digits";
    "၀၁၂၃၄၅၆၇၈၉",
```

```

        "_type";
        "numeric";
    }
    "mymrshan";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mymrtlng";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "newa";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "nkoo";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "olck";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "orya";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "osma";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "roman";
    {

```


[illegible]

```

        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
    "taluk";
    {
        "_digits";
        "താലൂക്കുകൾ",
        "_type";
        "numeric";
    }
    "tamil";
    {
        "_rules";
        "tamil",
        "_type";
        "algorithmic";
    }
    "tamildec";
    {
        "_digits";
        "0௧௨௩௪௫௬௭௮௯",
        "_type";
        "numeric";
    }
    "telu";
    {
        "_digits";
        "0౧౨౩౪౫౬౭౮౯",
        "_type";
        "numeric";
    }
    "thai";
    {
        "_digits";
        "๐๑๒๓๔๕๖๗๘๙",
        "_type";
        "numeric";
    }
    "tibet";
    {
        "_digits";
        "༠༡༢༣༤༥༦༧༨༩",
        "_type";
        "numeric";
    }
    "tirh";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
    "vaih";
    {

```

NUMBERS.JSON

Copyright © 2001 -2024 Syncfusion Inc.

```

        "100000-count-one": "000 Tausend",
        "100000-count-other": "000 Tausend",
        "1000000-count-one": "0 Million",
        "1000000-count-other": "0 Millionen",
        "10000000-count-one": "00 Millionen",
        "10000000-count-other": "00 Millionen",
        "100000000-count-one": "000 Millionen",
        "100000000-count-other": "000 Millionen",
        "1000000000-count-one": "0 Milliarde",
        "1000000000-count-other": "0 Milliarden",
        "10000000000-count-one": "00 Milliarden",
        "10000000000-count-other": "00 Milliarden",
        "100000000000-count-one": "000 Milliarden",
        "100000000000-count-other": "000 Milliarden",
        "1000000000000-count-one": "0 Billion",
        "1000000000000-count-other": "0 Billionen",
        "10000000000000-count-one": "00 Billionen",
        "10000000000000-count-other": "00 Billionen",
        "100000000000000-count-one": "000 Billionen",
        "100000000000000-count-other": "000 Billionen"
    }
},
"short": {
    "decimalFormat": {
        "1000-count-one": "0",
        "1000-count-other": "0",
        "10000-count-one": "0",
        "10000-count-other": "0",
        "100000-count-one": "0",
        "100000-count-other": "0",
        "1000000-count-one": "0 Mio'.'",
        "1000000-count-other": "0 Mio'.'",
        "10000000-count-one": "00 Mio'.'",
        "10000000-count-other": "00 Mio'.'",
        "100000000-count-one": "000 Mio'.'",
        "100000000-count-other": "000 Mio'.'",
        "1000000000-count-one": "0 Mrd'.'",
        "1000000000-count-other": "0 Mrd'.'",
        "10000000000-count-one": "00 Mrd'.'",
        "10000000000-count-other": "00 Mrd'.'",
        "100000000000-count-one": "000 Mrd'.'",
        "100000000000-count-other": "000 Mrd'.'",
        "1000000000000-count-one": "0 Bio'.'",
        "1000000000000-count-other": "0 Bio'.'",
        "10000000000000-count-one": "00 Bio'.'",
        "10000000000000-count-other": "00 Bio'.'",
        "100000000000000-count-one": "000 Bio'.'",
        "100000000000000-count-other": "000 Bio'.'"
    }
},
},
"scientificFormats-numberSystem-latn": {
    "standard": "#E0"
},
"percentFormats-numberSystem-latn": {
    "standard": "#,##0 %"
},
},

```

```

    "currencyFormats-numberSystem-latn": {
      "currencySpacing": {
        "beforeCurrency": {
          "currencyMatch": "[:^S:]",
          "surroundingMatch": "[:digit:]",
          "insertBetween": " "
        },
        "afterCurrency": {
          "currencyMatch": "[:^S:]",
          "surroundingMatch": "[:digit:]",
          "insertBetween": " "
        }
      },
      "standard": "#,##0.00 ¤",
      "accounting": "#,##0.00 ¤",
      "short": {
        "standard": {
          "1000-count-one": "0 Tsd'.' ¤",
          "1000-count-other": "0 Tsd'.' ¤",
          "10000-count-one": "00 Tsd'.' ¤",
          "10000-count-other": "00 Tsd'.' ¤",
          "100000-count-one": "000 Tsd'.' ¤",
          "100000-count-other": "000 Tsd'.' ¤",
          "1000000-count-one": "0 Mio'.' ¤",
          "1000000-count-other": "0 Mio'.' ¤",
          "10000000-count-one": "00 Mio'.' ¤",
          "10000000-count-other": "00 Mio'.' ¤",
          "100000000-count-one": "000 Mio'.' ¤",
          "100000000-count-other": "000 Mio'.' ¤",
          "1000000000-count-one": "0 Mrd'.' ¤",
          "1000000000-count-other": "0 Mrd'.' ¤",
          "10000000000-count-one": "00 Mrd'.' ¤",
          "10000000000-count-other": "00 Mrd'.' ¤",
          "100000000000-count-one": "000 Mrd'.' ¤",
          "100000000000-count-other": "000 Mrd'.' ¤",
          "1000000000000-count-one": "0 Bio'.' ¤",
          "1000000000000-count-other": "0 Bio'.' ¤",
          "10000000000000-count-one": "00 Bio'.' ¤",
          "10000000000000-count-other": "00 Bio'.' ¤",
          "100000000000000-count-one": "000 Bio'.' ¤",
          "100000000000000-count-other": "000 Bio'.' ¤"
        }
      },
      "unitPattern-count-one": "{0} {1}",
      "unitPattern-count-other": "{0} {1}"
    },
    "miscPatterns-numberSystem-latn": {
      "atLeast": "{0}+",
      "range": "{0}-{1}"
    },
    "minimalPairs": {
      "pluralMinimalPairs": "{0} Tag",
      "pluralMinimalPairs": "{0} Tage",
      "other": "{0}. Abzweigung nach rechts nehmen"
    }
  }
}

```

```
}
}
```

NUMBERS.JSX

```
{
  "main";
  {
    "de";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 13259 $",
          "_cldrVersion";
          "31";
        }
        "language";
        "de";
      }
    }
    "numbers";
    {
      "defaultNumberingSystem";
      "latn",
      "otherNumberingSystems";
      {
        "native";
        "latn";
      }
      "minimumGroupingDigits";
      "1",
      "symbols-numberSystem-latn";
      {
        "decimal";
        ",",
        "group";
        ".",
        "list";
        ";",
        "percentSign";
        "%",
        "plusSign";
        "+",
        "minusSign";
        "-",
        "exponential";
        "E",
        "superscriptingExponent";
        ". ",
        "perMille";
        "‰",
        "infinity";
        "∞",
        "nan";
      }
    }
  }
}
```

```

        "NaN",
        "timeSeparator";
        ":";
    }
    "decimalFormats-numberSystem-latn";
    {
        "standard";
        "#,##0.###",
        "long";
        {
            "decimalFormat";
            {
                "1000-count-one";
                "0 Tausend",
                "1000-count-other";
                "0 Tausend",
                "10000-count-one";
                "00 Tausend",
                "10000-count-other";
                "00 Tausend",
                "100000-count-one";
                "000 Tausend",
                "100000-count-other";
                "000 Tausend",
                "1000000-count-one";
                "0 Million",
                "1000000-count-other";
                "0 Millionen",
                "10000000-count-one";
                "00 Millionen",
                "10000000-count-other";
                "00 Millionen",
                "100000000-count-one";
                "000 Millionen",
                "100000000-count-other";
                "000 Millionen",
                "1000000000-count-one";
                "0 Milliarde",
                "1000000000-count-other";
                "0 Milliarden",
                "10000000000-count-one";
                "00 Milliarden",
                "10000000000-count-other";
                "00 Milliarden",
                "100000000000-count-one";
                "000 Milliarden",
                "100000000000-count-other";
                "000 Milliarden",
                "1000000000000-count-one";
                "0 Billion",
                "1000000000000-count-other";
                "0 Billionen",
                "10000000000000-count-one";
                "00 Billionen",
                "10000000000000-count-other";
                "00 Billionen",
                "100000000000000-count-one";
            }
        }
    }

```

```

        "000 Billionen",
        "1000000000000000-count-other";
        "000 Billionen";
    }
}
"short";
{
    "decimalFormat";
    {
        "1000-count-one";
        "0",
        "1000-count-other";
        "0",
        "10000-count-one";
        "0",
        "10000-count-other";
        "0",
        "100000-count-one";
        "0",
        "100000-count-other";
        "0",
        "1000000-count-one";
        "0 Mio'.'",
        "1000000-count-other";
        "0 Mio'.'",
        "10000000-count-one";
        "00 Mio'.'",
        "10000000-count-other";
        "00 Mio'.'",
        "100000000-count-one";
        "000 Mio'.'",
        "100000000-count-other";
        "000 Mio'.'",
        "1000000000-count-one";
        "0 Mrd'.'",
        "1000000000-count-other";
        "0 Mrd'.'",
        "10000000000-count-one";
        "00 Mrd'.'",
        "10000000000-count-other";
        "00 Mrd'.'",
        "100000000000-count-one";
        "000 Mrd'.'",
        "100000000000-count-other";
        "000 Mrd'.'",
        "1000000000000-count-one";
        "0 Bio'.'",
        "1000000000000-count-other";
        "0 Bio'.'",
        "10000000000000-count-one";
        "00 Bio'.'",
        "10000000000000-count-other";
        "00 Bio'.'",
        "100000000000000-count-one";
        "000 Bio'.'",
        "100000000000000-count-other";
        "000 Bio'.'";
    }
}

```



```

    }
  }
}
"scientificFormats-numberSystem-latn";
{
  "standard";
  "#E0";
}
"percentFormats-numberSystem-latn";
{
  "standard";
  "#,##0 %";
}
"currencyFormats-numberSystem-latn";
{
  "currencySpacing";
  {
    "beforeCurrency";
    {
      "currencyMatch";
      "[:^S:]",
      "surroundingMatch";
      "[:digit:]",
      "insertBetween";
      " ";
    }
    "afterCurrency";
    {
      "currencyMatch";
      "[:^S:]",
      "surroundingMatch";
      "[:digit:]",
      "insertBetween";
      " ";
    }
  }
}
"standard";
"#,##0.00 ¤",
"accounting";
"#,##0.00 ¤",
"short";
{
  "standard";
  {
    "1000-count-one";
    "0 Tsd'.' ¤",
    "1000-count-other";
    "0 Tsd'.' ¤",
    "10000-count-one";
    "00 Tsd'.' ¤",
    "10000-count-other";
    "00 Tsd'.' ¤",
    "100000-count-one";
    "000 Tsd'.' ¤",
    "100000-count-other";
    "000 Tsd'.' ¤",
    "1000000-count-one";
  }
}

```

```

        "0 Mio'.' ¤",
        "1000000-count-other";
        "0 Mio'.' ¤",
        "10000000-count-one";
        "00 Mio'.' ¤",
        "10000000-count-other";
        "00 Mio'.' ¤",
        "100000000-count-one";
        "000 Mio'.' ¤",
        "100000000-count-other";
        "000 Mio'.' ¤",
        "1000000000-count-one";
        "0 Mrd'.' ¤",
        "1000000000-count-other";
        "0 Mrd'.' ¤",
        "10000000000-count-one";
        "00 Mrd'.' ¤",
        "10000000000-count-other";
        "00 Mrd'.' ¤",
        "100000000000-count-one";
        "000 Mrd'.' ¤",
        "100000000000-count-other";
        "000 Mrd'.' ¤",
        "1000000000000-count-one";
        "0 Bio'.' ¤",
        "1000000000000-count-other";
        "0 Bio'.' ¤",
        "10000000000000-count-one";
        "00 Bio'.' ¤",
        "10000000000000-count-other";
        "00 Bio'.' ¤",
        "100000000000000-count-one";
        "000 Bio'.' ¤",
        "100000000000000-count-other";
        "000 Bio'.' ¤";
    }
}
"unitPattern-count-one";
"{0} {1}",
    "unitPattern-count-other";
"{0} {1}";
}
"miscPatterns-numberSystem-latn";
{
    "atLeast";
    "{0}+",
        "range";
    "{0}-{1}";
}
"minimalPairs";
{
    "pluralMinimalPairs";
    "{0} Tag",
        "pluralMinimalPairs";
    "{0} Tage",
        "other";
    "{0}. Abzweigung nach rechts nehmen";
}

```

```

    }
  }
}

```

TIMEZONENAMES.JSON

```

{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 12879 $",
          "_cldrVersion": "30.0.3"
        },
        "language": "de"
      },
      "dates": {
        "timeZoneNames": {
          "hourFormat": "+HH:mm;-HH:mm",
          "gmtFormat": "GMT{0}",
          "gmtZeroFormat": "GMT",
          "regionFormat": "{0} Zeit",
          "regionFormat-type-daylight": "{0} Sommerzeit",
          "regionFormat-type-standard": "{0} Normalzeit",
          "fallbackFormat": "{1} ({0})",
          "zone": {
            "America": {
              "Adak": {
                "exemplarCity": "Adak"
              },
              "Anchorage": {
                "exemplarCity": "Anchorage"
              },
              "Anguilla": {
                "exemplarCity": "Anguilla"
              },
              "Antigua": {
                "exemplarCity": "Antigua"
              },
              "Araguaina": {
                "exemplarCity": "Araguaina"
              },
              "Argentina": {
                "Rio_Gallegos": {
                  "exemplarCity": "Rio Gallegos"
                },
                "San_Juan": {
                  "exemplarCity": "San Juan"
                },
                "Ushuaia": {
                  "exemplarCity": "Ushuaia"
                },
                "La_Rioja": {
                  "exemplarCity": "La Rioja"
                }
              }
            }
          }
        }
      }
    }
  }
}

```

```
    },
    "San_Luis": {
      "exemplarCity": "San Luis"
    },
    "Salta": {
      "exemplarCity": "Salta"
    },
    "Tucuman": {
      "exemplarCity": "Tucuman"
    }
  },
  "Aruba": {
    "exemplarCity": "Aruba"
  },
  "Asuncion": {
    "exemplarCity": "Asunción"
  },
  "Bahia": {
    "exemplarCity": "Bahia"
  },
  "Bahia_Banderas": {
    "exemplarCity": "Bahia Banderas"
  },
  "Barbados": {
    "exemplarCity": "Barbados"
  },
  "Belem": {
    "exemplarCity": "Belem"
  },
  "Belize": {
    "exemplarCity": "Belize"
  },
  "Blanc-Sablon": {
    "exemplarCity": "Blanc-Sablon"
  },
  "Boa_Vista": {
    "exemplarCity": "Boa Vista"
  },
  "Bogota": {
    "exemplarCity": "Bogotá"
  },
  "Boise": {
    "exemplarCity": "Boise"
  },
  "Buenos_Aires": {
    "exemplarCity": "Buenos Aires"
  },
  "Cambridge_Bay": {
    "exemplarCity": "Cambridge Bay"
  },
  "Campo_Grande": {
    "exemplarCity": "Campo Grande"
  },
  "Cancun": {
    "exemplarCity": "Cancún"
  },
  "Caracas": {
```

```
    "exemplarCity": "Caracas"
  },
  "Catamarca": {
    "exemplarCity": "Catamarca"
  },
  "Cayenne": {
    "exemplarCity": "Cayenne"
  },
  "Cayman": {
    "exemplarCity": "Kaimaninseln"
  },
  "Chicago": {
    "exemplarCity": "Chicago"
  },
  "Chihuahua": {
    "exemplarCity": "Chihuahua"
  },
  "Coral_Harbour": {
    "exemplarCity": "Atikokan"
  },
  "Cordoba": {
    "exemplarCity": "Córdoba"
  },
  "Costa_Rica": {
    "exemplarCity": "Costa Rica"
  },
  "Creston": {
    "exemplarCity": "Creston"
  },
  "Cuiaba": {
    "exemplarCity": "Cuiaba"
  },
  "Curacao": {
    "exemplarCity": "Curaçao"
  },
  "Danmarkshavn": {
    "exemplarCity": "Danmarkshavn"
  },
  "Dawson": {
    "exemplarCity": "Dawson"
  },
  "Dawson_Creek": {
    "exemplarCity": "Dawson Creek"
  },
  "Denver": {
    "exemplarCity": "Denver"
  },
  "Detroit": {
    "exemplarCity": "Detroit"
  },
  "Dominica": {
    "exemplarCity": "Dominica"
  },
  "Edmonton": {
    "exemplarCity": "Edmonton"
  },
  "Eirunepe": {
```

```

        "exemplarCity": "Eirunepe"
    },
    "El_Salvador": {
        "exemplarCity": "El Salvador"
    },
    "Fort_Nelson": {
        "exemplarCity": "Fort Nelson"
    },
    "Fortaleza": {
        "exemplarCity": "Fortaleza"
    },
    "Glace_Bay": {
        "exemplarCity": "Glace Bay"
    },
    "Godthab": {
        "exemplarCity": "Nuuk"
    },
    "Goose_Bay": {
        "exemplarCity": "Goose Bay"
    },
    "Grand_Turk": {
        "exemplarCity": "Grand Turk"
    },
    "Grenada": {
        "exemplarCity": "Grenada"
    },
    "Guadeloupe": {
        "exemplarCity": "Guadeloupe"
    },
    "Guatemala": {
        "exemplarCity": "Guatemala"
    },
    "Guayaquil": {
        "exemplarCity": "Guayaquil"
    },
    "Guyana": {
        "exemplarCity": "Guyana"
    },
    "Halifax": {
        "exemplarCity": "Halifax"
    },
    "Havana": {
        "exemplarCity": "Havanna"
    },
    "Hermosillo": {
        "exemplarCity": "Hermosillo"
    },
    "Indiana": {
        "Vincennes": {
            "exemplarCity": "Vincennes, Indiana"
        },
        "Petersburg": {
            "exemplarCity": "Petersburg, Indiana"
        },
        "Tell_City": {
            "exemplarCity": "Tell City, Indiana"
        }
    },

```

```
    "Knox": {
      "exemplarCity": "Knox, Indiana"
    },
    "Winamac": {
      "exemplarCity": "Winamac, Indiana"
    },
    "Marengo": {
      "exemplarCity": "Marengo, Indiana"
    },
    "Vevay": {
      "exemplarCity": "Vevay, Indiana"
    }
  },
  "Indianapolis": {
    "exemplarCity": "Indianapolis"
  },
  "Inuvik": {
    "exemplarCity": "Inuvik"
  },
  "Iqaluit": {
    "exemplarCity": "Iqaluit"
  },
  "Jamaica": {
    "exemplarCity": "Jamaika"
  },
  "Jujuy": {
    "exemplarCity": "Jujuy"
  },
  "Juneau": {
    "exemplarCity": "Juneau"
  },
  "Kentucky": {
    "Monticello": {
      "exemplarCity": "Monticello, Kentucky"
    }
  },
  "Kralendijk": {
    "exemplarCity": "Kralendijk"
  },
  "La_Paz": {
    "exemplarCity": "La Paz"
  },
  "Lima": {
    "exemplarCity": "Lima"
  },
  "Los_Angeles": {
    "exemplarCity": "Los Angeles"
  },
  "Louisville": {
    "exemplarCity": "Louisville"
  },
  "Lower_Princes": {
    "exemplarCity": "Lower Prince's Quarter"
  },
  "Maceio": {
    "exemplarCity": "Maceio"
  },
}
```

```
"Managua": {
  "exemplarCity": "Managua"
},
"Manaus": {
  "exemplarCity": "Manaus"
},
"Marigot": {
  "exemplarCity": "Marigot"
},
"Martinique": {
  "exemplarCity": "Martinique"
},
"Matamoros": {
  "exemplarCity": "Matamoros"
},
"Mazatlan": {
  "exemplarCity": "Mazatlan"
},
"Mendoza": {
  "exemplarCity": "Mendoza"
},
"Menominee": {
  "exemplarCity": "Menominee"
},
"Merida": {
  "exemplarCity": "Merida"
},
"Metlakatla": {
  "exemplarCity": "Metlakatla"
},
"Mexico_City": {
  "exemplarCity": "Mexiko-Stadt"
},
"Miquelon": {
  "exemplarCity": "Miquelon"
},
"Moncton": {
  "exemplarCity": "Moncton"
},
"Monterrey": {
  "exemplarCity": "Monterrey"
},
"Montevideo": {
  "exemplarCity": "Montevideo"
},
"Montserrat": {
  "exemplarCity": "Montserrat"
},
"Nassau": {
  "exemplarCity": "Nassau"
},
"New_York": {
  "exemplarCity": "New York"
},
"Nipigon": {
  "exemplarCity": "Nipigon"
},
},
```



```
"Nome": {
  "exemplarCity": "Nome"
},
"Noronha": {
  "exemplarCity": "Noronha"
},
"North_Dakota": {
  "Beulah": {
    "exemplarCity": "Beulah, North Dakota"
  },
  "New_Salem": {
    "exemplarCity": "New Salem, North Dakota"
  },
  "Center": {
    "exemplarCity": "Center, North Dakota"
  }
},
"Ojinaga": {
  "exemplarCity": "Ojinaga"
},
"Panama": {
  "exemplarCity": "Panama"
},
"Pangnirtung": {
  "exemplarCity": "Pangnirtung"
},
"Paramaribo": {
  "exemplarCity": "Paramaribo"
},
"Phoenix": {
  "exemplarCity": "Phoenix"
},
"Port-au-Prince": {
  "exemplarCity": "Port-au-Prince"
},
"Port_of_Spain": {
  "exemplarCity": "Port of Spain"
},
"Porto_Velho": {
  "exemplarCity": "Porto Velho"
},
"Puerto_Rico": {
  "exemplarCity": "Puerto Rico"
},
"Rainy_River": {
  "exemplarCity": "Rainy River"
},
"Rankin_Inlet": {
  "exemplarCity": "Rankin Inlet"
},
"Recife": {
  "exemplarCity": "Recife"
},
"Regina": {
  "exemplarCity": "Regina"
},
"Resolute": {
```

```
    "exemplarCity": "Resolute"
  },
  "Rio_Branco": {
    "exemplarCity": "Rio Branco"
  },
  "Santa_Isabel": {
    "exemplarCity": "Santa Isabel"
  },
  "Santarem": {
    "exemplarCity": "Santarem"
  },
  "Santiago": {
    "exemplarCity": "Santiago"
  },
  "Santo_Domingo": {
    "exemplarCity": "Santo Domingo"
  },
  "Sao_Paulo": {
    "exemplarCity": "São Paulo"
  },
  "Scoresbysund": {
    "exemplarCity": "Ittoqqortoormiit"
  },
  "Sitka": {
    "exemplarCity": "Sitka"
  },
  "St_Barthelemy": {
    "exemplarCity": "Saint-Barthélemy"
  },
  "St_Johns": {
    "exemplarCity": "St. John's"
  },
  "St_Kitts": {
    "exemplarCity": "St. Kitts"
  },
  "St_Lucia": {
    "exemplarCity": "St. Lucia"
  },
  "St_Thomas": {
    "exemplarCity": "St. Thomas"
  },
  "St_Vincent": {
    "exemplarCity": "St. Vincent"
  },
  "Swift_Current": {
    "exemplarCity": "Swift Current"
  },
  "Tegucigalpa": {
    "exemplarCity": "Tegucigalpa"
  },
  "Thule": {
    "exemplarCity": "Thule"
  },
  "Thunder_Bay": {
    "exemplarCity": "Thunder Bay"
  },
  "Tijuana": {
```

```
        "exemplarCity": "Tijuana"
      },
      "Toronto": {
        "exemplarCity": "Toronto"
      },
      "Tortola": {
        "exemplarCity": "Tortola"
      },
      "Vancouver": {
        "exemplarCity": "Vancouver"
      },
      "Whitehorse": {
        "exemplarCity": "Whitehorse"
      },
      "Winnipeg": {
        "exemplarCity": "Winnipeg"
      },
      "Yakutat": {
        "exemplarCity": "Yakutat"
      },
      "Yellowknife": {
        "exemplarCity": "Yellowknife"
      }
    },
    "Atlantic": {
      "Azores": {
        "exemplarCity": "Azoren"
      },
      "Bermuda": {
        "exemplarCity": "Bermudas"
      },
      "Canary": {
        "exemplarCity": "Kanaren"
      },
      "Cape_Verde": {
        "exemplarCity": "Cabo Verde"
      },
      "Faeroe": {
        "exemplarCity": "Färöer"
      },
      "Madeira": {
        "exemplarCity": "Madeira"
      },
      "Reykjavik": {
        "exemplarCity": "Reykjavík"
      },
      "South_Georgia": {
        "exemplarCity": "Südgeorgien"
      },
      "St_Helena": {
        "exemplarCity": "St. Helena"
      },
      "Stanley": {
        "exemplarCity": "Stanley"
      }
    },
    "Europe": {
```

```
"Amsterdam": {
  "exemplarCity": "Amsterdam"
},
"Andorra": {
  "exemplarCity": "Andorra"
},
"Astrakhan": {
  "exemplarCity": "Astrachan"
},
"Athens": {
  "exemplarCity": "Athen"
},
"Belgrade": {
  "exemplarCity": "Belgrad"
},
"Berlin": {
  "exemplarCity": "Berlin"
},
"Bratislava": {
  "exemplarCity": "Bratislava"
},
"Brussels": {
  "exemplarCity": "Brüssel"
},
"Bucharest": {
  "exemplarCity": "Bukarest"
},
"Budapest": {
  "exemplarCity": "Budapest"
},
"Busingen": {
  "exemplarCity": "Büsingen"
},
"Chisinau": {
  "exemplarCity": "Kischinau"
},
"Copenhagen": {
  "exemplarCity": "Kopenhagen"
},
"Dublin": {
  "long": {
    "daylight": "Irische Sommerzeit"
  },
  "exemplarCity": "Dublin"
},
"Gibraltar": {
  "exemplarCity": "Gibraltar"
},
"Guernsey": {
  "exemplarCity": "Guernsey"
},
"Helsinki": {
  "exemplarCity": "Helsinki"
},
"Isle_of_Man": {
  "exemplarCity": "Isle of Man"
},
```

```
"Istanbul": {
  "exemplarCity": "Istanbul"
},
"Jersey": {
  "exemplarCity": "Jersey"
},
"Kaliningrad": {
  "exemplarCity": "Kaliningrad"
},
"Kiev": {
  "exemplarCity": "Kiew"
},
"Kirov": {
  "exemplarCity": "Kirow"
},
"Lisbon": {
  "exemplarCity": "Lissabon"
},
"Ljubljana": {
  "exemplarCity": "Ljubljana"
},
"London": {
  "long": {
    "daylight": "Britische Sommerzeit"
  },
  "exemplarCity": "London"
},
"Luxembourg": {
  "exemplarCity": "Luxemburg"
},
"Madrid": {
  "exemplarCity": "Madrid"
},
"Malta": {
  "exemplarCity": "Malta"
},
"Mariehamn": {
  "exemplarCity": "Mariehamn"
},
"Minsk": {
  "exemplarCity": "Minsk"
},
"Monaco": {
  "exemplarCity": "Monaco"
},
"Moscow": {
  "exemplarCity": "Moskau"
},
"Oslo": {
  "exemplarCity": "Oslo"
},
"Paris": {
  "exemplarCity": "Paris"
},
"Podgorica": {
  "exemplarCity": "Podgorica"
},
```

```
"Prague": {
  "exemplarCity": "Prag"
},
"Riga": {
  "exemplarCity": "Riga"
},
"Rome": {
  "exemplarCity": "Rom"
},
"Samara": {
  "exemplarCity": "Samara"
},
"San_Marino": {
  "exemplarCity": "San Marino"
},
"Sarajevo": {
  "exemplarCity": "Sarajevo"
},
"Simferopol": {
  "exemplarCity": "Simferopol"
},
"Skopje": {
  "exemplarCity": "Skopje"
},
"Sofia": {
  "exemplarCity": "Sofia"
},
"Stockholm": {
  "exemplarCity": "Stockholm"
},
"Tallinn": {
  "exemplarCity": "Tallinn"
},
"Tirane": {
  "exemplarCity": "Tirana"
},
"Ulyanovsk": {
  "exemplarCity": "Uljanowsk"
},
"Uzhgorod": {
  "exemplarCity": "Uschgorod"
},
"Vaduz": {
  "exemplarCity": "Vaduz"
},
"Vatican": {
  "exemplarCity": "Vatikan"
},
"Vienna": {
  "exemplarCity": "Wien"
},
"Vilnius": {
  "exemplarCity": "Vilnius"
},
"Volgograd": {
  "exemplarCity": "Wolgograd"
},
```

```
"Warsaw": {
  "exemplarCity": "Warschau"
},
"Zagreb": {
  "exemplarCity": "Zagreb"
},
"Zaporozhye": {
  "exemplarCity": "Saporischja"
},
"Zurich": {
  "exemplarCity": "Zürich"
}
},
"Africa": {
  "Abidjan": {
    "exemplarCity": "Abidjan"
  },
  "Accra": {
    "exemplarCity": "Accra"
  },
  "Addis_Ababa": {
    "exemplarCity": "Addis Abeba"
  },
  "Algiers": {
    "exemplarCity": "Algier"
  },
  "Asmera": {
    "exemplarCity": "Asmara"
  },
  "Bamako": {
    "exemplarCity": "Bamako"
  },
  "Bangui": {
    "exemplarCity": "Bangui"
  },
  "Banjul": {
    "exemplarCity": "Banjul"
  },
  "Bissau": {
    "exemplarCity": "Bissau"
  },
  "Blantyre": {
    "exemplarCity": "Blantyre"
  },
  "Brazzaville": {
    "exemplarCity": "Brazzaville"
  },
  "Bujumbura": {
    "exemplarCity": "Bujumbura"
  },
  "Cairo": {
    "exemplarCity": "Kairo"
  },
  "Casablanca": {
    "exemplarCity": "Casablanca"
  },
  "Ceuta": {
```

```
        "exemplarCity": "Ceuta"
    },
    "Conakry": {
        "exemplarCity": "Conakry"
    },
    "Dakar": {
        "exemplarCity": "Dakar"
    },
    "Dar_es_Salaam": {
        "exemplarCity": "Daressalam"
    },
    "Djibouti": {
        "exemplarCity": "Dschibuti"
    },
    "Douala": {
        "exemplarCity": "Douala"
    },
    "El_Aaiun": {
        "exemplarCity": "El Aaiún"
    },
    "Freetown": {
        "exemplarCity": "Freetown"
    },
    "Gaborone": {
        "exemplarCity": "Gaborone"
    },
    "Harare": {
        "exemplarCity": "Harare"
    },
    "Johannesburg": {
        "exemplarCity": "Johannesburg"
    },
    "Juba": {
        "exemplarCity": "Juba"
    },
    "Kampala": {
        "exemplarCity": "Kampala"
    },
    "Khartoum": {
        "exemplarCity": "Khartum"
    },
    "Kigali": {
        "exemplarCity": "Kigali"
    },
    "Kinshasa": {
        "exemplarCity": "Kinshasa"
    },
    "Lagos": {
        "exemplarCity": "Lagos"
    },
    "Libreville": {
        "exemplarCity": "Libreville"
    },
    "Lome": {
        "exemplarCity": "Lomé"
    },
    "Luanda": {
```



```
        "exemplarCity": "Luanda"
    },
    "Lubumbashi": {
        "exemplarCity": "Lubumbashi"
    },
    "Lusaka": {
        "exemplarCity": "Lusaka"
    },
    "Malabo": {
        "exemplarCity": "Malabo"
    },
    "Maputo": {
        "exemplarCity": "Maputo"
    },
    "Maseru": {
        "exemplarCity": "Maseru"
    },
    "Mbabane": {
        "exemplarCity": "Mbabane"
    },
    "Mogadishu": {
        "exemplarCity": "Mogadischu"
    },
    "Monrovia": {
        "exemplarCity": "Monrovia"
    },
    "Nairobi": {
        "exemplarCity": "Nairobi"
    },
    "Ndjamena": {
        "exemplarCity": "N' Djamena"
    },
    "Niamey": {
        "exemplarCity": "Niamey"
    },
    "Nouakchott": {
        "exemplarCity": "Nouakchott"
    },
    "Ouagadougou": {
        "exemplarCity": "Ouagadougou"
    },
    "Porto-Novo": {
        "exemplarCity": "Porto Novo"
    },
    "Sao_Tome": {
        "exemplarCity": "São Tomé"
    },
    "Tripoli": {
        "exemplarCity": "Tripolis"
    },
    "Tunis": {
        "exemplarCity": "Tunis"
    },
    "Windhoek": {
        "exemplarCity": "Windhoek"
    }
},
```

```
"Asia": {
  "Aden": {
    "exemplarCity": "Aden"
  },
  "Almaty": {
    "exemplarCity": "Almaty"
  },
  "Amman": {
    "exemplarCity": "Amman"
  },
  "Anadyr": {
    "exemplarCity": "Anadyr"
  },
  "Aqtau": {
    "exemplarCity": "Aqtau"
  },
  "Aqtobe": {
    "exemplarCity": "Aktobe"
  },
  "Ashgabat": {
    "exemplarCity": "Aşgabat"
  },
  "Baghdad": {
    "exemplarCity": "Bagdad"
  },
  "Bahrain": {
    "exemplarCity": "Bahrain"
  },
  "Baku": {
    "exemplarCity": "Baku"
  },
  "Bangkok": {
    "exemplarCity": "Bangkok"
  },
  "Barnaul": {
    "exemplarCity": "Barnaul"
  },
  "Beirut": {
    "exemplarCity": "Beirut"
  },
  "Bishkek": {
    "exemplarCity": "Bischkek"
  },
  "Brunei": {
    "exemplarCity": "Brunei"
  },
  "Calcutta": {
    "exemplarCity": "Kalkutta"
  },
  "Chita": {
    "exemplarCity": "Tschita"
  },
  "Choibalsan": {
    "exemplarCity": "Tschoibalsan"
  },
  "Colombo": {
    "exemplarCity": "Colombo"
  }
}
```

```
    },
    "Damascus": {
      "exemplarCity": "Damaskus"
    },
    "Dhaka": {
      "exemplarCity": "Dhaka"
    },
    "Dili": {
      "exemplarCity": "Dili"
    },
    "Dubai": {
      "exemplarCity": "Dubai"
    },
    "Dushanbe": {
      "exemplarCity": "Duschanbe"
    },
    "Gaza": {
      "exemplarCity": "Gaza"
    },
    "Hebron": {
      "exemplarCity": "Hebron"
    },
    "Hong_Kong": {
      "exemplarCity": "Hongkong"
    },
    "Hovd": {
      "exemplarCity": "Chowd"
    },
    "Irkutsk": {
      "exemplarCity": "Irkutsk"
    },
    "Jakarta": {
      "exemplarCity": "Jakarta"
    },
    "Jayapura": {
      "exemplarCity": "Jayapura"
    },
    "Jerusalem": {
      "exemplarCity": "Jerusalem"
    },
    "Kabul": {
      "exemplarCity": "Kabul"
    },
    "Kamchatka": {
      "exemplarCity": "Kamtschatka"
    },
    "Karachi": {
      "exemplarCity": "Karatschi"
    },
    "Katmandu": {
      "exemplarCity": "Kathmandu"
    },
    "Khandyga": {
      "exemplarCity": "Chandyga"
    },
    "Krasnoyarsk": {
      "exemplarCity": "Krasnojarsk"
    }
  }
```

```
    },
    "Kuala_Lumpur": {
      "exemplarCity": "Kuala Lumpur"
    },
    "Kuching": {
      "exemplarCity": "Kuching"
    },
    "Kuwait": {
      "exemplarCity": "Kuwait"
    },
    "Macau": {
      "exemplarCity": "Macao"
    },
    "Magadan": {
      "exemplarCity": "Magadan"
    },
    "Makassar": {
      "exemplarCity": "Makassar"
    },
    "Manila": {
      "exemplarCity": "Manila"
    },
    "Muscat": {
      "exemplarCity": "Maskat"
    },
    "Nicosia": {
      "exemplarCity": "Nikosia"
    },
    "Novokuznetsk": {
      "exemplarCity": "Nowokuznetsk"
    },
    "Novosibirsk": {
      "exemplarCity": "Nowosibirsk"
    },
    "Omsk": {
      "exemplarCity": "Omsk"
    },
    "Oral": {
      "exemplarCity": "Oral"
    },
    "Phnom_Penh": {
      "exemplarCity": "Phnom Penh"
    },
    "Pontianak": {
      "exemplarCity": "Pontianak"
    },
    "Pyongyang": {
      "exemplarCity": "Pjöngjang"
    },
    "Qatar": {
      "exemplarCity": "Katar"
    },
    "Qyzylorda": {
      "exemplarCity": "Qysylorda"
    },
    "Rangoon": {
      "exemplarCity": "Rangun"
    }
```

```
    },
    "Riyadh": {
      "exemplarCity": "Riad"
    },
    "Saigon": {
      "exemplarCity": "Ho-Chi-Minh-Stadt"
    },
    "Sakhalin": {
      "exemplarCity": "Sachalin"
    },
    "Samarkand": {
      "exemplarCity": "Samarkand"
    },
    "Seoul": {
      "exemplarCity": "Seoul"
    },
    "Shanghai": {
      "exemplarCity": "Shanghai"
    },
    "Singapore": {
      "exemplarCity": "Singapur"
    },
    "Srednekolymsk": {
      "exemplarCity": "Srednekolymsk"
    },
    "Taipei": {
      "exemplarCity": "Taipeh"
    },
    "Tashkent": {
      "exemplarCity": "Taschkent"
    },
    "Tbilisi": {
      "exemplarCity": "Tiflis"
    },
    "Tehran": {
      "exemplarCity": "Teheran"
    },
    "Thimphu": {
      "exemplarCity": "Thimphu"
    },
    "Tokyo": {
      "exemplarCity": "Tokio"
    },
    "Tomsk": {
      "exemplarCity": "Tomsk"
    },
    "Ulaanbaatar": {
      "exemplarCity": "Ulaanbaatar"
    },
    "Urumqi": {
      "exemplarCity": "Ürümqi"
    },
    "Ust-Nera": {
      "exemplarCity": "Ust-Nera"
    },
    "Vientiane": {
      "exemplarCity": "Vientiane"
    }
  }
```

```

    },
    "Vladivostok": {
      "exemplarCity": "Wladiwostok"
    },
    "Yakutsk": {
      "exemplarCity": "Jakutsk"
    },
    "Yekaterinburg": {
      "exemplarCity": "Jekaterinburg"
    },
    "Yerevan": {
      "exemplarCity": "Eriwan"
    }
  },
  "Indian": {
    "Antananarivo": {
      "exemplarCity": "Antananarivo"
    },
    "Chagos": {
      "exemplarCity": "Chagos"
    },
    "Christmas": {
      "exemplarCity": "Weihnachtsinsel"
    },
    "Cocos": {
      "exemplarCity": "Cocos"
    },
    "Comoro": {
      "exemplarCity": "Komoren"
    },
    "Kerguelen": {
      "exemplarCity": "Kerguelen"
    },
    "Mahe": {
      "exemplarCity": "Mahe"
    },
    "Maldives": {
      "exemplarCity": "Malediven"
    },
    "Mauritius": {
      "exemplarCity": "Mauritius"
    },
    "Mayotte": {
      "exemplarCity": "Mayotte"
    },
    "Reunion": {
      "exemplarCity": "Réunion"
    }
  },
  "Australia": {
    "Adelaide": {
      "exemplarCity": "Adelaide"
    },
    "Brisbane": {
      "exemplarCity": "Brisbane"
    },
    "Broken_Hill": {

```

```
        "exemplarCity": "Broken Hill"
      },
      "Currie": {
        "exemplarCity": "Currie"
      },
      "Darwin": {
        "exemplarCity": "Darwin"
      },
      "Eucla": {
        "exemplarCity": "Eucla"
      },
      "Hobart": {
        "exemplarCity": "Hobart"
      },
      "Lindeman": {
        "exemplarCity": "Lindeman"
      },
      "Lord_Howe": {
        "exemplarCity": "Lord Howe"
      },
      "Melbourne": {
        "exemplarCity": "Melbourne"
      },
      "Perth": {
        "exemplarCity": "Perth"
      },
      "Sydney": {
        "exemplarCity": "Sydney"
      }
    },
    "Pacific": {
      "Apia": {
        "exemplarCity": "Apia"
      },
      "Auckland": {
        "exemplarCity": "Auckland"
      },
      "Bougainville": {
        "exemplarCity": "Bougainville"
      },
      "Chatham": {
        "exemplarCity": "Chatham"
      },
      "Easter": {
        "exemplarCity": "Osterinsel"
      },
      "Efate": {
        "exemplarCity": "Efate"
      },
      "Enderbury": {
        "exemplarCity": "Enderbury"
      },
      "Fakaofu": {
        "exemplarCity": "Fakaofu"
      },
      "Fiji": {
        "exemplarCity": "Fidschi"
      }
    }
  }
}
```

```
    },  
    "Funafuti": {  
      "exemplarCity": "Funafuti"  
    },  
    "Galapagos": {  
      "exemplarCity": "Galapagos"  
    },  
    "Gambier": {  
      "exemplarCity": "Gambier"  
    },  
    "Guadalcanal": {  
      "exemplarCity": "Guadalcanal"  
    },  
    "Guam": {  
      "exemplarCity": "Guam"  
    },  
    "Honolulu": {  
      "exemplarCity": "Honolulu"  
    },  
    "Johnston": {  
      "exemplarCity": "Johnston"  
    },  
    "Kiritimati": {  
      "exemplarCity": "Kiritimati"  
    },  
    "Kosrae": {  
      "exemplarCity": "Kosrae"  
    },  
    "Kwajalein": {  
      "exemplarCity": "Kwajalein"  
    },  
    "Majuro": {  
      "exemplarCity": "Majuro"  
    },  
    "Marquesas": {  
      "exemplarCity": "Marquesas"  
    },  
    "Midway": {  
      "exemplarCity": "Midway"  
    },  
    "Nauru": {  
      "exemplarCity": "Nauru"  
    },  
    "Niue": {  
      "exemplarCity": "Niue"  
    },  
    "Norfolk": {  
      "exemplarCity": "Norfolk"  
    },  
    "Noumea": {  
      "exemplarCity": "Noumea"  
    },  
    "Pago_Pago": {  
      "exemplarCity": "Pago Pago"  
    },  
    "Palau": {  
      "exemplarCity": "Palau"
```



```
    },
    "Pitcairn": {
      "exemplarCity": "Pitcairn"
    },
    "Ponape": {
      "exemplarCity": "Pohnpei"
    },
    "Port_Moresby": {
      "exemplarCity": "Port Moresby"
    },
    "Rarotonga": {
      "exemplarCity": "Rarotonga"
    },
    "Saipan": {
      "exemplarCity": "Saipan"
    },
    "Tahiti": {
      "exemplarCity": "Tahiti"
    },
    "Tarawa": {
      "exemplarCity": "Tarawa"
    },
    "Tongatapu": {
      "exemplarCity": "Tongatapu"
    },
    "Truk": {
      "exemplarCity": "Chuuk"
    },
    "Wake": {
      "exemplarCity": "Wake"
    },
    "Wallis": {
      "exemplarCity": "Wallis"
    }
  },
  "Arctic": {
    "Longyearbyen": {
      "exemplarCity": "Longyearbyen"
    }
  },
  "Antarctica": {
    "Casey": {
      "exemplarCity": "Casey"
    },
    "Davis": {
      "exemplarCity": "Davis"
    },
    "DumontDUrville": {
      "exemplarCity": "Dumont d'Urville"
    },
    "Macquarie": {
      "exemplarCity": "Macquarie"
    },
    "Mawson": {
      "exemplarCity": "Mawson"
    },
    "McMurdo": {
```

```
        "exemplarCity": "McMurdo"
      },
      "Palmer": {
        "exemplarCity": "Palmer"
      },
      "Rothera": {
        "exemplarCity": "Rothera"
      },
      "Syowa": {
        "exemplarCity": "Syowa"
      },
      "Troll": {
        "exemplarCity": "Troll"
      },
      "Vostok": {
        "exemplarCity": "Wostok"
      }
    },
    "Etc": {
      "GMT": {
        "exemplarCity": "GMT"
      },
      "GMT1": {
        "exemplarCity": "GMT+1"
      },
      "GMT10": {
        "exemplarCity": "GMT+10"
      },
      "GMT11": {
        "exemplarCity": "GMT+11"
      },
      "GMT12": {
        "exemplarCity": "GMT+12"
      },
      "GMT2": {
        "exemplarCity": "GMT+2"
      },
      "GMT3": {
        "exemplarCity": "GMT+3"
      },
      "GMT4": {
        "exemplarCity": "GMT+4"
      },
      "GMT5": {
        "exemplarCity": "GMT+5"
      },
      "GMT6": {
        "exemplarCity": "GMT+6"
      },
      "GMT7": {
        "exemplarCity": "GMT+7"
      },
      "GMT8": {
        "exemplarCity": "GMT+8"
      },
      "GMT9": {
        "exemplarCity": "GMT+9"
      }
    }
  }
}
```

```

    },
    "GMT-1": {
      "exemplarCity": "GMT-1"
    },
    "GMT-10": {
      "exemplarCity": "GMT-10"
    },
    "GMT-11": {
      "exemplarCity": "GMT-11"
    },
    "GMT-12": {
      "exemplarCity": "GMT-12"
    },
    "GMT-13": {
      "exemplarCity": "GMT-13"
    },
    "GMT-14": {
      "exemplarCity": "GMT-14"
    },
    "GMT-2": {
      "exemplarCity": "GMT-2"
    },
    "GMT-3": {
      "exemplarCity": "GMT-3"
    },
    "GMT-4": {
      "exemplarCity": "GMT-4"
    },
    "GMT-5": {
      "exemplarCity": "GMT-5"
    },
    "GMT-6": {
      "exemplarCity": "GMT-6"
    },
    "GMT-7": {
      "exemplarCity": "GMT-7"
    },
    "GMT-8": {
      "exemplarCity": "GMT-8"
    },
    "GMT-9": {
      "exemplarCity": "GMT-9"
    },
    "Unknown": {
      "exemplarCity": "Unbekannt"
    }
  },
  "metazone": {
    "Acre": {
      "long": {
        "generic": "Acre-Zeit",
        "standard": "Acre-Normalzeit",
        "daylight": "Acre-Sommerzeit"
      }
    },
    "Afghanistan": {

```

```

        "long": {
          "standard": "Afghanistan-Zeit"
        }
      },
      "Africa_Central": {
        "long": {
          "standard": "Zentralafrikanische Zeit"
        }
      },
      "Africa_Eastern": {
        "long": {
          "standard": "Ostafrikanische Zeit"
        }
      },
      "Africa_Southern": {
        "long": {
          "standard": "Südafrikanische Zeit"
        }
      },
      "Africa_Western": {
        "long": {
          "generic": "Westafrikanische Zeit",
          "standard": "Westafrikanische Normalzeit",
          "daylight": "Westafrikanische Sommerzeit"
        }
      },
      "Alaska": {
        "long": {
          "generic": "Alaska-Zeit",
          "standard": "Alaska-Normalzeit",
          "daylight": "Alaska-Sommerzeit"
        }
      },
      "Almaty": {
        "long": {
          "generic": "Almaty-Zeit",
          "standard": "Almaty-Normalzeit",
          "daylight": "Almaty-Sommerzeit"
        }
      },
      "Amazon": {
        "long": {
          "generic": "Amazonas-Zeit",
          "standard": "Amazonas-Normalzeit",
          "daylight": "Amazonas-Sommerzeit"
        }
      },
      "America_Central": {
        "long": {
          "generic": "Nordamerikanische Inlandzeit",
          "standard": "Nordamerikanische Inland-Normalzeit",
          "daylight": "Nordamerikanische Inland-Sommerzeit"
        }
      },
      "America_Eastern": {
        "long": {
          "generic": "Nordamerikanische Ostküstenzeit",

```

```

        "standard": "Nordamerikanische Ostküsten-Normalzeit",
        "daylight": "Nordamerikanische Ostküsten-Sommerzeit"
    },
    },
    "America_Mountain": {
        "long": {
            "generic": "Rocky-Mountain-Zeit",
            "standard": "Rocky Mountain-Normalzeit",
            "daylight": "Rocky-Mountain-Sommerzeit"
        }
    },
    "America_Pacific": {
        "long": {
            "generic": "Nordamerikanische Westküstenzeit",
            "standard": "Nordamerikanische Westküsten-Normalzeit",
            "daylight": "Nordamerikanische Westküsten-Sommerzeit"
        }
    },
    "Anadyr": {
        "long": {
            "generic": "Anadyr Zeit",
            "standard": "Anadyr Normalzeit",
            "daylight": "Anadyr Sommerzeit"
        }
    },
    "Apia": {
        "long": {
            "generic": "Apia-Zeit",
            "standard": "Apia-Normalzeit",
            "daylight": "Apia-Sommerzeit"
        }
    },
    "Aqttau": {
        "long": {
            "generic": "Aqttau-Zeit",
            "standard": "Aqttau-Normalzeit",
            "daylight": "Aqttau-Sommerzeit"
        }
    },
    "Aqtobe": {
        "long": {
            "generic": "Aqtöbe-Zeit",
            "standard": "Aqtöbe-Normalzeit",
            "daylight": "Aqtöbe-Sommerzeit"
        }
    },
    "Arabian": {
        "long": {
            "generic": "Arabische Zeit",
            "standard": "Arabische Normalzeit",
            "daylight": "Arabische Sommerzeit"
        }
    },
    "Argentina": {
        "long": {
            "generic": "Argentinische Zeit",
            "standard": "Argentinische Normalzeit",

```

```

        "daylight": "Argentinische Sommerzeit"
    },
    },
    "Argentina_Western": {
        "long": {
            "generic": "Westargentinische Zeit",
            "standard": "Westargentinische Normalzeit",
            "daylight": "Westargentinische Sommerzeit"
        }
    },
    "Armenia": {
        "long": {
            "generic": "Armenische Zeit",
            "standard": "Armenische Normalzeit",
            "daylight": "Armenische Sommerzeit"
        }
    },
    "Atlantic": {
        "long": {
            "generic": "Atlantik-Zeit",
            "standard": "Atlantik-Normalzeit",
            "daylight": "Atlantik-Sommerzeit"
        }
    },
    "Australia_Central": {
        "long": {
            "generic": "Zentralaustralische Zeit",
            "standard": "Zentralaustralische Normalzeit",
            "daylight": "Zentralaustralische Sommerzeit"
        }
    },
    "Australia_CentralWestern": {
        "long": {
            "generic": "Zentral-/Westaustralische Zeit",
            "standard": "Zentral-/Westaustralische Normalzeit",
            "daylight": "Zentral-/Westaustralische Sommerzeit"
        }
    },
    "Australia_Eastern": {
        "long": {
            "generic": "Ostaustralische Zeit",
            "standard": "Ostaustralische Normalzeit",
            "daylight": "Ostaustralische Sommerzeit"
        }
    },
    "Australia_Western": {
        "long": {
            "generic": "Westaustralische Zeit",
            "standard": "Westaustralische Normalzeit",
            "daylight": "Westaustralische Sommerzeit"
        }
    },
    "Azerbaijan": {
        "long": {
            "generic": "Aserbaidshanische Zeit",
            "standard": "Aserbeidschanische Normalzeit",
            "daylight": "Aserbaidshanische Sommerzeit"
        }
    }
}

```

```

    }
  },
  "Azores": {
    "long": {
      "generic": "Azoren-Zeit",
      "standard": "Azoren-Normalzeit",
      "daylight": "Azoren-Sommerzeit"
    }
  },
  "Bangladesh": {
    "long": {
      "generic": "Bangladesch-Zeit",
      "standard": "Bangladesch-Normalzeit",
      "daylight": "Bangladesch-Sommerzeit"
    }
  },
  "Bhutan": {
    "long": {
      "standard": "Bhutan-Zeit"
    }
  },
  "Bolivia": {
    "long": {
      "standard": "Bolivianische Zeit"
    }
  },
  "Brasilia": {
    "long": {
      "generic": "Brasília-Zeit",
      "standard": "Brasília-Normalzeit",
      "daylight": "Brasília-Sommerzeit"
    }
  },
  "Brunei": {
    "long": {
      "standard": "Brunei-Zeit"
    }
  },
  "Cape_Verde": {
    "long": {
      "generic": "Cabo-Verde-Zeit",
      "standard": "Cabo-Verde-Normalzeit",
      "daylight": "Cabo-Verde-Sommerzeit"
    }
  },
  "Casey": {
    "long": {
      "standard": "Casey-Zeit"
    }
  },
  "Chamorro": {
    "long": {
      "standard": "Chamorro-Zeit"
    }
  },
  "Chatham": {
    "long": {

```

```

        "generic": "Chatham-Zeit",
        "standard": "Chatham-Normalzeit",
        "daylight": "Chatham-Sommerzeit"
    },
    },
    "Chile": {
        "long": {
            "generic": "Chilenische Zeit",
            "standard": "Chilenische Normalzeit",
            "daylight": "Chilenische Sommerzeit"
        }
    },
    },
    "China": {
        "long": {
            "generic": "Chinesische Zeit",
            "standard": "Chinesische Normalzeit",
            "daylight": "Chinesische Sommerzeit"
        }
    },
    },
    "Choibalsan": {
        "long": {
            "generic": "Tschoibalsan-Zeit",
            "standard": "Tschoibalsan-Normalzeit",
            "daylight": "Tschoibalsan-Sommerzeit"
        }
    },
    },
    "Christmas": {
        "long": {
            "standard": "Weihnachtsinsel-Zeit"
        }
    },
    },
    "Cocos": {
        "long": {
            "standard": "Kokosinseln-Zeit"
        }
    },
    },
    "Colombia": {
        "long": {
            "generic": "Kolumbianische Zeit",
            "standard": "Kolumbianische Normalzeit",
            "daylight": "Kolumbianische Sommerzeit"
        }
    },
    },
    "Cook": {
        "long": {
            "generic": "Cookinseln-Zeit",
            "standard": "Cookinseln-Normalzeit",
            "daylight": "Cookinseln-Sommerzeit"
        }
    },
    },
    "Cuba": {
        "long": {
            "generic": "Kubanische Zeit",
            "standard": "Kubanische Normalzeit",
            "daylight": "Kubanische Sommerzeit"
        }
    },
    },
    },

```



```

    "Davis": {
      "long": {
        "standard": "Davis-Zeit"
      }
    },
    "DumontDUrville": {
      "long": {
        "standard": "Dumont-d'Urville-Zeit"
      }
    },
    "East_Timor": {
      "long": {
        "standard": "Osttimor-Zeit"
      }
    },
    "Easter": {
      "long": {
        "generic": "Osterinsel-Zeit",
        "standard": "Osterinsel-Normalzeit",
        "daylight": "Osterinsel-Sommerzeit"
      }
    },
    "Ecuador": {
      "long": {
        "standard": "Ecuadorianische Zeit"
      }
    },
    "Europe_Central": {
      "long": {
        "generic": "Mitteleuropäische Zeit",
        "standard": "Mitteleuropäische Normalzeit",
        "daylight": "Mitteleuropäische Sommerzeit"
      },
      "short": {
        "generic": "MEZ",
        "standard": "MEZ",
        "daylight": "MESZ"
      }
    },
    "Europe_Eastern": {
      "long": {
        "generic": "Osteuropäische Zeit",
        "standard": "Osteuropäische Normalzeit",
        "daylight": "Osteuropäische Sommerzeit"
      },
      "short": {
        "generic": "OEZ",
        "standard": "OEZ",
        "daylight": "OESZ"
      }
    },
    "Europe_Further_Eastern": {
      "long": {
        "standard": "Kaliningrader Zeit"
      }
    },
    "Europe_Western": {

```

```

    "long": {
      "generic": "Westeuropäische Zeit",
      "standard": "Westeuropäische Normalzeit",
      "daylight": "Westeuropäische Sommerzeit"
    },
    "short": {
      "generic": "WEZ",
      "standard": "WEZ",
      "daylight": "WESZ"
    }
  },
  "Falkland": {
    "long": {
      "generic": "Falklandinseln-Zeit",
      "standard": "Falklandinseln-Normalzeit",
      "daylight": "Falklandinseln-Sommerzeit"
    }
  },
  "Fiji": {
    "long": {
      "generic": "Fidschi-Zeit",
      "standard": "Fidschi-Normalzeit",
      "daylight": "Fidschi-Sommerzeit"
    }
  },
  "French_Guiana": {
    "long": {
      "standard": "Französisch-Guayana-Zeit"
    }
  },
  "French_Southern": {
    "long": {
      "standard": "Französische Süd- und Antarktisgebiete-Zeit"
    }
  },
  "Galapagos": {
    "long": {
      "standard": "Galapagos-Zeit"
    }
  },
  "Gambier": {
    "long": {
      "standard": "Gambier-Zeit"
    }
  },
  "Georgia": {
    "long": {
      "generic": "Georgische Zeit",
      "standard": "Georgische Normalzeit",
      "daylight": "Georgische Sommerzeit"
    }
  },
  "Gilbert_Islands": {
    "long": {
      "standard": "Gilbert-Inseln-Zeit"
    }
  },

```

```

"GMT": {
  "long": {
    "standard": "Mittlere Greenwich-Zeit"
  }
},
"Greenland_Eastern": {
  "long": {
    "generic": "Ostgrönland-Zeit",
    "standard": "Ostgrönland-Normalzeit",
    "daylight": "Ostgrönland-Sommerzeit"
  }
},
"Greenland_Western": {
  "long": {
    "generic": "Westgrönland-Zeit",
    "standard": "Westgrönland-Normalzeit",
    "daylight": "Westgrönland-Sommerzeit"
  }
},
"Guam": {
  "long": {
    "standard": "Guam-Zeit"
  }
},
"Gulf": {
  "long": {
    "standard": "Golf-Zeit"
  }
},
"Guyana": {
  "long": {
    "standard": "Guyana-Zeit"
  }
},
"Hawaii_Aleutian": {
  "long": {
    "generic": "Hawaii-Aleuten-Zeit",
    "standard": "Hawaii-Aleuten-Normalzeit",
    "daylight": "Hawaii-Aleuten-Sommerzeit"
  }
},
"Hong_Kong": {
  "long": {
    "generic": "Hongkong-Zeit",
    "standard": "Hongkong-Normalzeit",
    "daylight": "Hongkong-Sommerzeit"
  }
},
"Hovd": {
  "long": {
    "generic": "Chowd-Zeit",
    "standard": "Chowd-Normalzeit",
    "daylight": "Chowd-Sommerzeit"
  }
},
"India": {
  "long": {

```

```

        "standard": "Indische Zeit"
    },
    },
    "Indian_Ocean": {
        "long": {
            "standard": "Indischer Ozean-Zeit"
        }
    },
    "Indochina": {
        "long": {
            "standard": "Indochina-Zeit"
        }
    },
    "Indonesia_Central": {
        "long": {
            "standard": "Zentralindonesische Zeit"
        }
    },
    "Indonesia_Eastern": {
        "long": {
            "standard": "Ostindonesische Zeit"
        }
    },
    "Indonesia_Western": {
        "long": {
            "standard": "Westindonesische Zeit"
        }
    },
    "Iran": {
        "long": {
            "generic": "Iranische Zeit",
            "standard": "Iranische Normalzeit",
            "daylight": "Iranische Sommerzeit"
        }
    },
    "Irkutsk": {
        "long": {
            "generic": "Irkutsk-Zeit",
            "standard": "Irkutsk-Normalzeit",
            "daylight": "Irkutsk-Sommerzeit"
        }
    },
    "Israel": {
        "long": {
            "generic": "Israelische Zeit",
            "standard": "Israelische Normalzeit",
            "daylight": "Israelische Sommerzeit"
        }
    },
    "Japan": {
        "long": {
            "generic": "Japanische Zeit",
            "standard": "Japanische Normalzeit",
            "daylight": "Japanische Sommerzeit"
        }
    },
    "Kamchatka": {

```

```

        "long": {
            "generic": "Kamtschatka-Zeit",
            "standard": "Kamtschatka-Normalzeit",
            "daylight": "Kamtschatka-Sommerzeit"
        }
    },
    "Kazakhstan_Eastern": {
        "long": {
            "standard": "Ostkasachische Zeit"
        }
    },
    "Kazakhstan_Western": {
        "long": {
            "standard": "Westkasachische Zeit"
        }
    },
    "Korea": {
        "long": {
            "generic": "Koreanische Zeit",
            "standard": "Koreanische Normalzeit",
            "daylight": "Koreanische Sommerzeit"
        }
    },
    "Kosrae": {
        "long": {
            "standard": "Kosrae-Zeit"
        }
    },
    "Krasnoyarsk": {
        "long": {
            "generic": "Krasnojarsk-Zeit",
            "standard": "Krasnojarsk-Normalzeit",
            "daylight": "Krasnojarsk-Sommerzeit"
        }
    },
    "Kyrgystan": {
        "long": {
            "standard": "Kirgisistan-Zeit"
        }
    },
    "Lanka": {
        "long": {
            "standard": "Sri-Lanka-Zeit"
        }
    },
    "Line_Islands": {
        "long": {
            "standard": "Linieninseln-Zeit"
        }
    },
    "Lord_Howe": {
        "long": {
            "generic": "Lord-Howe-Zeit",
            "standard": "Lord-Howe-Normalzeit",
            "daylight": "Lord-Howe-Sommerzeit"
        }
    },
    },

```

```
"Macau": {
  "long": {
    "generic": "Macau-Zeit",
    "standard": "Macau-Normalzeit",
    "daylight": "Macau-Sommerzeit"
  }
},
"Macquarie": {
  "long": {
    "standard": "Macquarieinsel-Zeit"
  }
},
"Magadan": {
  "long": {
    "generic": "Magadan-Zeit",
    "standard": "Magadan-Normalzeit",
    "daylight": "Magadan-Sommerzeit"
  }
},
"Malaysia": {
  "long": {
    "standard": "Malaysische Zeit"
  }
},
"Maldives": {
  "long": {
    "standard": "Malediven-Zeit"
  }
},
"Marquesas": {
  "long": {
    "standard": "Marquesas-Zeit"
  }
},
"Marshall_Islands": {
  "long": {
    "standard": "Marshallinseln-Zeit"
  }
},
"Mauritius": {
  "long": {
    "generic": "Mauritius-Zeit",
    "standard": "Mauritius-Normalzeit",
    "daylight": "Mauritius-Sommerzeit"
  }
},
"Mawson": {
  "long": {
    "standard": "Mawson-Zeit"
  }
},
"Mexico_Northwest": {
  "long": {
    "generic": "Mexiko Nordwestliche Zone-Zeit",
    "standard": "Mexiko Nordwestliche Zone-Normalzeit",
    "daylight": "Mexiko Nordwestliche Zone-Sommerzeit"
  }
}
```

```
    },  
    "Mexico_Pacific": {  
      "long": {  
        "generic": "Mexiko Pazifikzone-Zeit",  
        "standard": "Mexiko Pazifikzone-Normalzeit",  
        "daylight": "Mexiko Pazifikzone-Sommerzeit"  
      }  
    },  
    "Mongolia": {  
      "long": {  
        "generic": "Ulaanbaatar-Zeit",  
        "standard": "Ulaanbaatar-Normalzeit",  
        "daylight": "Ulaanbaatar-Sommerzeit"  
      }  
    },  
    "Moscow": {  
      "long": {  
        "generic": "Moskauer Zeit",  
        "standard": "Moskauer Normalzeit",  
        "daylight": "Moskauer Sommerzeit"  
      }  
    },  
    "Myanmar": {  
      "long": {  
        "standard": "Myanmar-Zeit"  
      }  
    },  
    "Nauru": {  
      "long": {  
        "standard": "Nauru-Zeit"  
      }  
    },  
    "Nepal": {  
      "long": {  
        "standard": "Nepalesische Zeit"  
      }  
    },  
    "New_Caledonia": {  
      "long": {  
        "generic": "Neukaledonische Zeit",  
        "standard": "Neukaledonische Normalzeit",  
        "daylight": "Neukaledonische Sommerzeit"  
      }  
    },  
    "New_Zealand": {  
      "long": {  
        "generic": "Neuseeland-Zeit",  
        "standard": "Neuseeland-Normalzeit",  
        "daylight": "Neuseeland-Sommerzeit"  
      }  
    },  
    "Newfoundland": {  
      "long": {  
        "generic": "Neufundland-Zeit",  
        "standard": "Neufundland-Normalzeit",  
        "daylight": "Neufundland-Sommerzeit"  
      }  
    }  
  }  
}
```

```

    },
    "Niue": {
      "long": {
        "standard": "Niue-Zeit"
      }
    },
    "Norfolk": {
      "long": {
        "standard": "Norfolkinsel-Zeit"
      }
    },
    "Noronha": {
      "long": {
        "generic": "Fernando de Noronha-Zeit",
        "standard": "Fernando de Noronha-Normalzeit",
        "daylight": "Fernando de Noronha-Sommerzeit"
      }
    },
    "North_Mariana": {
      "long": {
        "standard": "Nördliche-Marianen-Zeit"
      }
    },
    "Novosibirsk": {
      "long": {
        "generic": "Nowosibirsk-Zeit",
        "standard": "Nowosibirsk-Normalzeit",
        "daylight": "Nowosibirsk-Sommerzeit"
      }
    },
    "Omsk": {
      "long": {
        "generic": "Omsk-Zeit",
        "standard": "Omsk-Normalzeit",
        "daylight": "Omsk-Sommerzeit"
      }
    },
    "Pakistan": {
      "long": {
        "generic": "Pakistanische Zeit",
        "standard": "Pakistanische Normalzeit",
        "daylight": "Pakistanische Sommerzeit"
      }
    },
    "Palau": {
      "long": {
        "standard": "Palau-Zeit"
      }
    },
    "Papua_New_Guinea": {
      "long": {
        "standard": "Papua-Neuguinea-Zeit"
      }
    },
    "Paraguay": {
      "long": {
        "generic": "Paraguayanische Zeit",

```



```

        "standard": "Paraguayische Normalzeit",
        "daylight": "Paraguayische Sommerzeit"
    },
    "Peru": {
        "long": {
            "generic": "Peruanische Zeit",
            "standard": "Peruanische Normalzeit",
            "daylight": "Peruanische Sommerzeit"
        }
    },
    "Philippines": {
        "long": {
            "generic": "Philippinische Zeit",
            "standard": "Philippinische Normalzeit",
            "daylight": "Philippinische Sommerzeit"
        }
    },
    "Phoenix_Islands": {
        "long": {
            "standard": "Phoenixinseln-Zeit"
        }
    },
    "Pierre_Miquelon": {
        "long": {
            "generic": "Saint-Pierre-und-Miquelon-Zeit",
            "standard": "Saint-Pierre-und-Miquelon-Normalzeit",
            "daylight": "Saint-Pierre-und-Miquelon-Sommerzeit"
        }
    },
    "Pitcairn": {
        "long": {
            "standard": "Pitcairnsinseln-Zeit"
        }
    },
    "Ponape": {
        "long": {
            "standard": "Ponape-Zeit"
        }
    },
    "Pyongyang": {
        "long": {
            "standard": "Pjöngjang-Zeit"
        }
    },
    "Qyzylorda": {
        "long": {
            "generic": "Quysylorda-Zeit",
            "standard": "Quysylorda-Normalzeit",
            "daylight": "Qysylorda-Sommerzeit"
        }
    },
    "Reunion": {
        "long": {
            "standard": "Réunion-Zeit"
        }
    },
    },

```

```
"Rothera": {
  "long": {
    "standard": "Rothera-Zeit"
  }
},
"Sakhalin": {
  "long": {
    "generic": "Sachalin-Zeit",
    "standard": "Sachalin-Normalzeit",
    "daylight": "Sachalin-Sommerzeit"
  }
},
"Samara": {
  "long": {
    "generic": "Samara-Zeit",
    "standard": "Samara-Normalzeit",
    "daylight": "Samara-Sommerzeit"
  }
},
"Samoa": {
  "long": {
    "generic": "Samoa-Zeit",
    "standard": "Samoa-Normalzeit",
    "daylight": "Samoa-Sommerzeit"
  }
},
"Seychelles": {
  "long": {
    "standard": "Seychellen-Zeit"
  }
},
"Singapore": {
  "long": {
    "standard": "Singapur-Zeit"
  }
},
"Solomon": {
  "long": {
    "standard": "Salomoninseln-Zeit"
  }
},
"South_Georgia": {
  "long": {
    "standard": "Südgeorgische Zeit"
  }
},
"Suriname": {
  "long": {
    "standard": "Suriname-Zeit"
  }
},
"Syowa": {
  "long": {
    "standard": "Syowa-Zeit"
  }
},
"Tahiti": {
```

```

        "long": {
            "standard": "Tahiti-Zeit"
        }
    },
    "Taipei": {
        "long": {
            "generic": "Taipeh-Zeit",
            "standard": "Taipeh-Normalzeit",
            "daylight": "Taipeh-Sommerzeit"
        }
    },
    "Tajikistan": {
        "long": {
            "standard": "Tadschikistan-Zeit"
        }
    },
    "Tokelau": {
        "long": {
            "standard": "Tokelau-Zeit"
        }
    },
    "Tonga": {
        "long": {
            "generic": "Tonganische Zeit",
            "standard": "Tonganische Normalzeit",
            "daylight": "Tonganische Sommerzeit"
        }
    },
    "Truk": {
        "long": {
            "standard": "Chuuk-Zeit"
        }
    },
    "Turkmenistan": {
        "long": {
            "generic": "Turkmenistan-Zeit",
            "standard": "Turkmenistan-Normalzeit",
            "daylight": "Turkmenistan-Sommerzeit"
        }
    },
    "Tuvalu": {
        "long": {
            "standard": "Tuvalu-Zeit"
        }
    },
    "Uruguay": {
        "long": {
            "generic": "Uruguayanische Zeit",
            "standard": "Uruguayanische Normalzeit",
            "daylight": "Uruguayanische Sommerzeit"
        }
    },
    "Uzbekistan": {
        "long": {
            "generic": "Usbekistan-Zeit",
            "standard": "Usbekistan-Normalzeit",
            "daylight": "Usbekistan-Sommerzeit"
        }
    }
}

```

```

    }
  },
  "Vanuatu": {
    "long": {
      "generic": "Vanuatu-Zeit",
      "standard": "Vanuatu-Normalzeit",
      "daylight": "Vanuatu-Sommerzeit"
    }
  },
  "Venezuela": {
    "long": {
      "standard": "Venezuela-Zeit"
    }
  },
  "Vladivostok": {
    "long": {
      "generic": "Wladiwostok-Zeit",
      "standard": "Wladiwostok-Normalzeit",
      "daylight": "Wladiwostok-Sommerzeit"
    }
  },
  "Volgograd": {
    "long": {
      "generic": "Wolgograd-Zeit",
      "standard": "Wolgograd-Normalzeit",
      "daylight": "Wolgograd-Sommerzeit"
    }
  },
  "Vostok": {
    "long": {
      "standard": "Wostok-Zeit"
    }
  },
  "Wake": {
    "long": {
      "standard": "Wake-Insel-Zeit"
    }
  },
  "Wallis": {
    "long": {
      "standard": "Wallis-und-Futuna-Zeit"
    }
  },
  "Yakutsk": {
    "long": {
      "generic": "Jakutsk-Zeit",
      "standard": "Jakutsk-Normalzeit",
      "daylight": "Jakutsk-Sommerzeit"
    }
  },
  "Yekaterinburg": {
    "long": {
      "generic": "Jekaterinburg-Zeit",
      "standard": "Jekaterinburg-Normalzeit",
      "daylight": "Jekaterinburg-Sommerzeit"
    }
  }
}

```

```

    }
  }
}
}
}

```

TIMEZONENAMES.JSX

```

{
  "main";
  {
    "de";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 12879 $",
          "_cldrVersion";
          "30.0.3";
        }
        "language";
        "de";
      }
      "dates";
      {
        "timeZoneNames";
        {
          "hourFormat";
          "+HH:mm;-HH:mm",
          "gmtFormat";
          "GMT{0}",
          "gmtZeroFormat";
          "GMT",
          "regionFormat";
          "{0} Zeit",
          "regionFormat-type-daylight";
          "{0} Sommerzeit",
          "regionFormat-type-standard";
          "{0} Normalzeit",
          "fallbackFormat";
          "{1} ({0})",
          "zone";
          {
            "America";
            {
              "Adak";
              {
                "exemplarCity";
                "Adak";
              }
              "Anchorage";
              {
                "exemplarCity";

```

```

        "Anchorage";
    }
    "Anguilla";
    {
        "exemplarCity";
        "Anguilla";
    }
    "Antigua";
    {
        "exemplarCity";
        "Antigua";
    }
    "Araguaina";
    {
        "exemplarCity";
        "Araguaina";
    }
    "Argentina";
    {
        "Rio_Gallegos";
        {
            "exemplarCity";
            "Rio Gallegos";
        }
        "San_Juan";
        {
            "exemplarCity";
            "San Juan";
        }
        "Ushuaia";
        {
            "exemplarCity";
            "Ushuaia";
        }
        "La_Rioja";
        {
            "exemplarCity";
            "La Rioja";
        }
        "San_Luis";
        {
            "exemplarCity";
            "San Luis";
        }
        "Salta";
        {
            "exemplarCity";
            "Salta";
        }
        "Tucuman";
        {
            "exemplarCity";
            "Tucuman";
        }
    }
    "Aruba";
    {

```

```
        "exemplarCity";
        "Aruba";
    }
    "Asuncion";
    {
        "exemplarCity";
        "Asunción";
    }
    "Bahia";
    {
        "exemplarCity";
        "Bahia";
    }
    "Bahia_Banderas";
    {
        "exemplarCity";
        "Bahia Banderas";
    }
    "Barbados";
    {
        "exemplarCity";
        "Barbados";
    }
    "Belem";
    {
        "exemplarCity";
        "Belem";
    }
    "Belize";
    {
        "exemplarCity";
        "Belize";
    }
    "Blanc-Sablon";
    {
        "exemplarCity";
        "Blanc-Sablon";
    }
    "Boa_Vista";
    {
        "exemplarCity";
        "Boa Vista";
    }
    "Bogota";
    {
        "exemplarCity";
        "Bogotá";
    }
    "Boise";
    {
        "exemplarCity";
        "Boise";
    }
    "Buenos_Aires";
    {
        "exemplarCity";
        "Buenos Aires";
    }
```

```
}
"Cambridge_Bay";
{
  "exemplarCity";
  "Cambridge Bay";
}
"Campo_Grande";
{
  "exemplarCity";
  "Campo Grande";
}
"Cancun";
{
  "exemplarCity";
  "Cancún";
}
"Caracas";
{
  "exemplarCity";
  "Caracas";
}
"Catamarca";
{
  "exemplarCity";
  "Catamarca";
}
"Cayenne";
{
  "exemplarCity";
  "Cayenne";
}
"Cayman";
{
  "exemplarCity";
  "Kaimaninseln";
}
"Chicago";
{
  "exemplarCity";
  "Chicago";
}
"Chihuahua";
{
  "exemplarCity";
  "Chihuahua";
}
"Coral_Harbour";
{
  "exemplarCity";
  "Atikokan";
}
"Cordoba";
{
  "exemplarCity";
  "Córdoba";
}
"Costa_Rica";
```



```
{
    "exemplarCity";
    "Costa Rica";
}
"Creston";
{
    "exemplarCity";
    "Creston";
}
"Cuiaba";
{
    "exemplarCity";
    "Cuiaba";
}
"Curacao";
{
    "exemplarCity";
    "Curaçao";
}
"Danmarkshavn";
{
    "exemplarCity";
    "Danmarkshavn";
}
"Dawson";
{
    "exemplarCity";
    "Dawson";
}
"Dawson_Creek";
{
    "exemplarCity";
    "Dawson Creek";
}
"Denver";
{
    "exemplarCity";
    "Denver";
}
"Detroit";
{
    "exemplarCity";
    "Detroit";
}
"Dominica";
{
    "exemplarCity";
    "Dominica";
}
"Edmonton";
{
    "exemplarCity";
    "Edmonton";
}
"Eirunepe";
{
    "exemplarCity";
```

```
        "Eirunepe";
    }
    "El_Salvador";
    {
        "exemplarCity";
        "El Salvador";
    }
    "Fort_Nelson";
    {
        "exemplarCity";
        "Fort Nelson";
    }
    "Fortaleza";
    {
        "exemplarCity";
        "Fortaleza";
    }
    "Glace_Bay";
    {
        "exemplarCity";
        "Glace Bay";
    }
    "Godthab";
    {
        "exemplarCity";
        "Nuuk";
    }
    "Goose_Bay";
    {
        "exemplarCity";
        "Goose Bay";
    }
    "Grand_Turk";
    {
        "exemplarCity";
        "Grand Turk";
    }
    "Grenada";
    {
        "exemplarCity";
        "Grenada";
    }
    "Guadeloupe";
    {
        "exemplarCity";
        "Guadeloupe";
    }
    "Guatemala";
    {
        "exemplarCity";
        "Guatemala";
    }
    "Guayaquil";
    {
        "exemplarCity";
        "Guayaquil";
    }
}
```

```
"Guyana";
{
  "exemplarCity";
  "Guyana";
}
"Halifax";
{
  "exemplarCity";
  "Halifax";
}
"Havana";
{
  "exemplarCity";
  "Havanna";
}
"Hermosillo";
{
  "exemplarCity";
  "Hermosillo";
}
"Indiana";
{
  "Vincennes";
  {
    "exemplarCity";
    "Vincennes, Indiana";
  }
  "Petersburg";
  {
    "exemplarCity";
    "Petersburg, Indiana";
  }
  "Tell_City";
  {
    "exemplarCity";
    "Tell City, Indiana";
  }
  "Knox";
  {
    "exemplarCity";
    "Knox, Indiana";
  }
  "Winamac";
  {
    "exemplarCity";
    "Winamac, Indiana";
  }
  "Marengo";
  {
    "exemplarCity";
    "Marengo, Indiana";
  }
  "Vevay";
  {
    "exemplarCity";
    "Vevay, Indiana";
  }
}
```

```
}
"Indianapolis";
{
  "exemplarCity";
  "Indianapolis";
}
"Inuvik";
{
  "exemplarCity";
  "Inuvik";
}
"Iqaluit";
{
  "exemplarCity";
  "Iqaluit";
}
"Jamaica";
{
  "exemplarCity";
  "Jamaika";
}
"Jujuy";
{
  "exemplarCity";
  "Jujuy";
}
"Juneau";
{
  "exemplarCity";
  "Juneau";
}
"Kentucky";
{
  "Monticello";
  {
    "exemplarCity";
    "Monticello, Kentucky";
  }
}
"Kralendijk";
{
  "exemplarCity";
  "Kralendijk";
}
"La_Paz";
{
  "exemplarCity";
  "La Paz";
}
"Lima";
{
  "exemplarCity";
  "Lima";
}
"Los_Angeles";
{
  "exemplarCity";
```

```

        "Los Angeles";
    }
    "Louisville";
    {
        "exemplarCity";
        "Louisville";
    }
    "Lower_Princes";
    {
        "exemplarCity";
        "Lower Prince's Quarter";
    }
    "Maceio";
    {
        "exemplarCity";
        "Maceio";
    }
    "Managua";
    {
        "exemplarCity";
        "Managua";
    }
    "Manaus";
    {
        "exemplarCity";
        "Manaus";
    }
    "Marigot";
    {
        "exemplarCity";
        "Marigot";
    }
    "Martinique";
    {
        "exemplarCity";
        "Martinique";
    }
    "Matamoros";
    {
        "exemplarCity";
        "Matamoros";
    }
    "Mazatlan";
    {
        "exemplarCity";
        "Mazatlan";
    }
    "Mendoza";
    {
        "exemplarCity";
        "Mendoza";
    }
    "Menominee";
    {
        "exemplarCity";
        "Menominee";
    }
}

```

```
"Merida";
{
  "exemplarCity";
  "Merida";
}
"Metlakatla";
{
  "exemplarCity";
  "Metlakatla";
}
"Mexico_City";
{
  "exemplarCity";
  "Mexiko-Stadt";
}
"Miquelon";
{
  "exemplarCity";
  "Miquelon";
}
"Moncton";
{
  "exemplarCity";
  "Moncton";
}
"Monterrey";
{
  "exemplarCity";
  "Monterrey";
}
"Montevideo";
{
  "exemplarCity";
  "Montevideo";
}
"Montserrat";
{
  "exemplarCity";
  "Montserrat";
}
"Nassau";
{
  "exemplarCity";
  "Nassau";
}
"New_York";
{
  "exemplarCity";
  "New York";
}
"Nipigon";
{
  "exemplarCity";
  "Nipigon";
}
"Nome";
{
```

```

        "exemplarCity";
        "Nome";
    }
    "Noronha";
    {
        "exemplarCity";
        "Noronha";
    }
    "North_Dakota";
    {
        "Beulah";
        {
            "exemplarCity";
            "Beulah, North Dakota";
        }
        "New_Salem";
        {
            "exemplarCity";
            "New Salem, North Dakota";
        }
        "Center";
        {
            "exemplarCity";
            "Center, North Dakota";
        }
    }
    "Ojinaga";
    {
        "exemplarCity";
        "Ojinaga";
    }
    "Panama";
    {
        "exemplarCity";
        "Panama";
    }
    "Pangnirtung";
    {
        "exemplarCity";
        "Pangnirtung";
    }
    "Paramaribo";
    {
        "exemplarCity";
        "Paramaribo";
    }
    "Phoenix";
    {
        "exemplarCity";
        "Phoenix";
    }
    "Port-au-Prince";
    {
        "exemplarCity";
        "Port-au-Prince";
    }
    "Port_of_Spain";

```

```
{
    "exemplarCity";
    "Port of Spain";
}
"Porto_Velho";
{
    "exemplarCity";
    "Porto Velho";
}
"Puerto_Rico";
{
    "exemplarCity";
    "Puerto Rico";
}
"Rainy_River";
{
    "exemplarCity";
    "Rainy River";
}
"Rankin_Inlet";
{
    "exemplarCity";
    "Rankin Inlet";
}
"Recife";
{
    "exemplarCity";
    "Recife";
}
"Regina";
{
    "exemplarCity";
    "Regina";
}
"Resolute";
{
    "exemplarCity";
    "Resolute";
}
"Rio_Branco";
{
    "exemplarCity";
    "Rio Branco";
}
"Santa_Isabel";
{
    "exemplarCity";
    "Santa Isabel";
}
"Santarem";
{
    "exemplarCity";
    "Santarem";
}
"Santiago";
{
    "exemplarCity";
```



```

        "Santiago";
    }
    "Santo_Domingo";
    {
        "exemplarCity";
        "Santo Domingo";
    }
    "Sao_Paulo";
    {
        "exemplarCity";
        "São Paulo";
    }
    "Scoresbysund";
    {
        "exemplarCity";
        "Ittoqqortoormiit";
    }
    "Sitka";
    {
        "exemplarCity";
        "Sitka";
    }
    "St_Barthelemy";
    {
        "exemplarCity";
        "Saint-Barthélemy";
    }
    "St_Johns";
    {
        "exemplarCity";
        "St. John's";
    }
    "St_Kitts";
    {
        "exemplarCity";
        "St. Kitts";
    }
    "St_Lucia";
    {
        "exemplarCity";
        "St. Lucia";
    }
    "St_Thomas";
    {
        "exemplarCity";
        "St. Thomas";
    }
    "St_Vincent";
    {
        "exemplarCity";
        "St. Vincent";
    }
    "Swift_Current";
    {
        "exemplarCity";
        "Swift Current";
    }
}

```

```
"Tegucigalpa";
{
  "exemplarCity";
  "Tegucigalpa";
}
"Thule";
{
  "exemplarCity";
  "Thule";
}
"Thunder_Bay";
{
  "exemplarCity";
  "Thunder Bay";
}
"Tijuana";
{
  "exemplarCity";
  "Tijuana";
}
"Toronto";
{
  "exemplarCity";
  "Toronto";
}
"Tortola";
{
  "exemplarCity";
  "Tortola";
}
"Vancouver";
{
  "exemplarCity";
  "Vancouver";
}
"Whitehorse";
{
  "exemplarCity";
  "Whitehorse";
}
"Winnipeg";
{
  "exemplarCity";
  "Winnipeg";
}
"Yakutat";
{
  "exemplarCity";
  "Yakutat";
}
"Yellowknife";
{
  "exemplarCity";
  "Yellowknife";
}
}
"Atlantic";
```

```

    {
      "Azores";
      {
        "exemplarCity";
        "Azoren";
      }
      "Bermuda";
      {
        "exemplarCity";
        "Bermudas";
      }
      "Canary";
      {
        "exemplarCity";
        "Kanaren";
      }
      "Cape_Verde";
      {
        "exemplarCity";
        "Cabo Verde";
      }
      "Faeroe";
      {
        "exemplarCity";
        "Färöer";
      }
      "Madeira";
      {
        "exemplarCity";
        "Madeira";
      }
      "Reykjavik";
      {
        "exemplarCity";
        "Reykjavík";
      }
      "South_Georgia";
      {
        "exemplarCity";
        "Südgeorgien";
      }
      "St_Helena";
      {
        "exemplarCity";
        "St. Helena";
      }
      "Stanley";
      {
        "exemplarCity";
        "Stanley";
      }
    }
    "Europe";
    {
      "Amsterdam";
      {
        "exemplarCity";

```

```
        "Amsterdam";
    }
    "Andorra";
    {
        "exemplarCity";
        "Andorra";
    }
    "Astrakhan";
    {
        "exemplarCity";
        "Astrachan";
    }
    "Athens";
    {
        "exemplarCity";
        "Athen";
    }
    "Belgrade";
    {
        "exemplarCity";
        "Belgrad";
    }
    "Berlin";
    {
        "exemplarCity";
        "Berlin";
    }
    "Bratislava";
    {
        "exemplarCity";
        "Bratislava";
    }
    "Brussels";
    {
        "exemplarCity";
        "Brüssel";
    }
    "Bucharest";
    {
        "exemplarCity";
        "Bukarest";
    }
    "Budapest";
    {
        "exemplarCity";
        "Budapest";
    }
    "Busingen";
    {
        "exemplarCity";
        "Büsingen";
    }
    "Chisinau";
    {
        "exemplarCity";
        "Kischinau";
    }
}
```

```
"Copenhagen";
{
  "exemplarCity";
  "Kopenhagen";
}
"Dublin";
{
  "long";
  {
    "daylight";
    "Irische Sommerzeit";
  }
  "exemplarCity";
  "Dublin";
}
"Gibraltar";
{
  "exemplarCity";
  "Gibraltar";
}
"Guernsey";
{
  "exemplarCity";
  "Guernsey";
}
"Helsinki";
{
  "exemplarCity";
  "Helsinki";
}
"Isle_of_Man";
{
  "exemplarCity";
  "Isle of Man";
}
"Istanbul";
{
  "exemplarCity";
  "Istanbul";
}
"Jersey";
{
  "exemplarCity";
  "Jersey";
}
"Kaliningrad";
{
  "exemplarCity";
  "Kaliningrad";
}
"Kiev";
{
  "exemplarCity";
  "Kiew";
}
"Kirov";
{
```

```
        "exemplarCity";
        "Kirow";
    }
    "Lisbon";
    {
        "exemplarCity";
        "Lissabon";
    }
    "Ljubljana";
    {
        "exemplarCity";
        "Ljubljana";
    }
    "London";
    {
        "long";
        {
            "daylight";
            "Britische Sommerzeit";
        }
        "exemplarCity";
        "London";
    }
    "Luxembourg";
    {
        "exemplarCity";
        "Luxemburg";
    }
    "Madrid";
    {
        "exemplarCity";
        "Madrid";
    }
    "Malta";
    {
        "exemplarCity";
        "Malta";
    }
    "Mariehamn";
    {
        "exemplarCity";
        "Mariehamn";
    }
    "Minsk";
    {
        "exemplarCity";
        "Minsk";
    }
    "Monaco";
    {
        "exemplarCity";
        "Monaco";
    }
    "Moscow";
    {
        "exemplarCity";
        "Moskau";
    }
```

```
}
"Oslo";
{
  "exemplarCity";
  "Oslo";
}
"Paris";
{
  "exemplarCity";
  "Paris";
}
"Podgorica";
{
  "exemplarCity";
  "Podgorica";
}
"Prague";
{
  "exemplarCity";
  "Prag";
}
"Riga";
{
  "exemplarCity";
  "Riga";
}
"Rome";
{
  "exemplarCity";
  "Rom";
}
"Samara";
{
  "exemplarCity";
  "Samara";
}
"San_Marino";
{
  "exemplarCity";
  "San Marino";
}
"Sarajevo";
{
  "exemplarCity";
  "Sarajevo";
}
"Simferopol";
{
  "exemplarCity";
  "Simferopol";
}
"Skopje";
{
  "exemplarCity";
  "Skopje";
}
"Sofia";
```

```
{
    "exemplarCity";
    "Sofia";
}
"Stockholm";
{
    "exemplarCity";
    "Stockholm";
}
"Tallinn";
{
    "exemplarCity";
    "Tallinn";
}
"Tirane";
{
    "exemplarCity";
    "Tirana";
}
"Ulyanovsk";
{
    "exemplarCity";
    "Uljanowsk";
}
"Uzhgorod";
{
    "exemplarCity";
    "Uschgorod";
}
"Vaduz";
{
    "exemplarCity";
    "Vaduz";
}
"Vatican";
{
    "exemplarCity";
    "Vatikan";
}
"Vienna";
{
    "exemplarCity";
    "Wien";
}
"Vilnius";
{
    "exemplarCity";
    "Vilnius";
}
"Volgograd";
{
    "exemplarCity";
    "Wolgograd";
}
"Warsaw";
{
    "exemplarCity";
```



```

        "Warschau";
    }
    "Zagreb";
    {
        "exemplarCity";
        "Zagreb";
    }
    "Zaporozhye";
    {
        "exemplarCity";
        "Saporischja";
    }
    "Zurich";
    {
        "exemplarCity";
        "Zürich";
    }
}
"Africa";
{
    "Abidjan";
    {
        "exemplarCity";
        "Abidjan";
    }
    "Accra";
    {
        "exemplarCity";
        "Accra";
    }
    "Addis_Ababa";
    {
        "exemplarCity";
        "Addis Abeba";
    }
    "Algiers";
    {
        "exemplarCity";
        "Algier";
    }
    "Asmera";
    {
        "exemplarCity";
        "Asmara";
    }
    "Bamako";
    {
        "exemplarCity";
        "Bamako";
    }
    "Bangui";
    {
        "exemplarCity";
        "Bangui";
    }
    "Banjul";
    {

```

```
        "exemplarCity";
        "Banjul";
    }
    "Bissau";
    {
        "exemplarCity";
        "Bissau";
    }
    "Blantyre";
    {
        "exemplarCity";
        "Blantyre";
    }
    "Brazzaville";
    {
        "exemplarCity";
        "Brazzaville";
    }
    "Bujumbura";
    {
        "exemplarCity";
        "Bujumbura";
    }
    "Cairo";
    {
        "exemplarCity";
        "Kairo";
    }
    "Casablanca";
    {
        "exemplarCity";
        "Casablanca";
    }
    "Ceuta";
    {
        "exemplarCity";
        "Ceuta";
    }
    "Conakry";
    {
        "exemplarCity";
        "Conakry";
    }
    "Dakar";
    {
        "exemplarCity";
        "Dakar";
    }
    "Dar_es_Salaam";
    {
        "exemplarCity";
        "Daressalam";
    }
    "Djibouti";
    {
        "exemplarCity";
        "Dschibuti";
    }
```

```
}
"Douala";
{
  "exemplarCity";
  "Douala";
}
"El_Aaiun";
{
  "exemplarCity";
  "El Aaiún";
}
"Freetown";
{
  "exemplarCity";
  "Freetown";
}
"Gaborone";
{
  "exemplarCity";
  "Gaborone";
}
"Harare";
{
  "exemplarCity";
  "Harare";
}
"Johannesburg";
{
  "exemplarCity";
  "Johannesburg";
}
"Juba";
{
  "exemplarCity";
  "Juba";
}
"Kampala";
{
  "exemplarCity";
  "Kampala";
}
"Khartoum";
{
  "exemplarCity";
  "Khartum";
}
"Kigali";
{
  "exemplarCity";
  "Kigali";
}
"Kinshasa";
{
  "exemplarCity";
  "Kinshasa";
}
"Lagos";
```

```
{
    "exemplarCity";
    "Lagos";
}
"Libreville";
{
    "exemplarCity";
    "Libreville";
}
"Lome";
{
    "exemplarCity";
    "Lomé";
}
"Luanda";
{
    "exemplarCity";
    "Luanda";
}
"Lubumbashi";
{
    "exemplarCity";
    "Lubumbashi";
}
"Lusaka";
{
    "exemplarCity";
    "Lusaka";
}
"Malabo";
{
    "exemplarCity";
    "Malabo";
}
"Maputo";
{
    "exemplarCity";
    "Maputo";
}
"Maseru";
{
    "exemplarCity";
    "Maseru";
}
"Mbabane";
{
    "exemplarCity";
    "Mbabane";
}
"Mogadishu";
{
    "exemplarCity";
    "Mogadischu";
}
"Monrovia";
{
    "exemplarCity";
```

```
        "Monrovia";
    }
    "Nairobi";
    {
        "exemplarCity";
        "Nairobi";
    }
    "Ndjamena";
    {
        "exemplarCity";
        "N' Djamena";
    }
    "Niamey";
    {
        "exemplarCity";
        "Niamey";
    }
    "Nouakchott";
    {
        "exemplarCity";
        "Nouakchott";
    }
    "Ouagadougou";
    {
        "exemplarCity";
        "Ouagadougou";
    }
    "Porto-Novo";
    {
        "exemplarCity";
        "Porto Novo";
    }
    "Sao_Tome";
    {
        "exemplarCity";
        "São Tomé";
    }
    "Tripoli";
    {
        "exemplarCity";
        "Tripolis";
    }
    "Tunis";
    {
        "exemplarCity";
        "Tunis";
    }
    "Windhoek";
    {
        "exemplarCity";
        "Windhoek";
    }
}
"Asia";
{
    "Aden";
    {
```

```
        "exemplarCity";
        "Aden";
    }
    "Almaty";
    {
        "exemplarCity";
        "Almaty";
    }
    "Amman";
    {
        "exemplarCity";
        "Amman";
    }
    "Anadyr";
    {
        "exemplarCity";
        "Anadyr";
    }
    "Aqtau";
    {
        "exemplarCity";
        "Aqtau";
    }
    "Aqtobe";
    {
        "exemplarCity";
        "Aktobe";
    }
    "Ashgabat";
    {
        "exemplarCity";
        "Aşgabat";
    }
    "Baghdad";
    {
        "exemplarCity";
        "Bagdad";
    }
    "Bahrain";
    {
        "exemplarCity";
        "Bahrain";
    }
    "Baku";
    {
        "exemplarCity";
        "Baku";
    }
    "Bangkok";
    {
        "exemplarCity";
        "Bangkok";
    }
    "Barnaul";
    {
        "exemplarCity";
        "Barnaul";
    }
```

```
}
"Beirut";
{
  "exemplarCity";
  "Beirut";
}
"Bishkek";
{
  "exemplarCity";
  "Bischkek";
}
"Brunei";
{
  "exemplarCity";
  "Brunei";
}
"Calcutta";
{
  "exemplarCity";
  "Kalkutta";
}
"Chita";
{
  "exemplarCity";
  "Tschita";
}
"Choibalsan";
{
  "exemplarCity";
  "Tschoibalsan";
}
"Colombo";
{
  "exemplarCity";
  "Colombo";
}
"Damascus";
{
  "exemplarCity";
  "Damaskus";
}
"Dhaka";
{
  "exemplarCity";
  "Dhaka";
}
"Dili";
{
  "exemplarCity";
  "Dili";
}
"Dubai";
{
  "exemplarCity";
  "Dubai";
}
"Dushanbe";
```

```
{
    "exemplarCity";
    "Duschanbe";
}
"Gaza";
{
    "exemplarCity";
    "Gaza";
}
"Hebron";
{
    "exemplarCity";
    "Hebron";
}
"Hong_Kong";
{
    "exemplarCity";
    "Hongkong";
}
"Hovd";
{
    "exemplarCity";
    "Chowd";
}
"Irkutsk";
{
    "exemplarCity";
    "Irkutsk";
}
"Jakarta";
{
    "exemplarCity";
    "Jakarta";
}
"Jayapura";
{
    "exemplarCity";
    "Jayapura";
}
"Jerusalem";
{
    "exemplarCity";
    "Jerusalem";
}
"Kabul";
{
    "exemplarCity";
    "Kabul";
}
"Kamchatka";
{
    "exemplarCity";
    "Kamtschatka";
}
"Karachi";
{
    "exemplarCity";
```



```
        "Karatschi";
    }
    "Katmandu";
    {
        "exemplarCity";
        "Kathmandu";
    }
    "Khandyga";
    {
        "exemplarCity";
        "Chandyga";
    }
    "Krasnoyarsk";
    {
        "exemplarCity";
        "Krasnojarsk";
    }
    "Kuala_Lumpur";
    {
        "exemplarCity";
        "Kuala Lumpur";
    }
    "Kuching";
    {
        "exemplarCity";
        "Kuching";
    }
    "Kuwait";
    {
        "exemplarCity";
        "Kuwait";
    }
    "Macau";
    {
        "exemplarCity";
        "Macao";
    }
    "Magadan";
    {
        "exemplarCity";
        "Magadan";
    }
    "Makassar";
    {
        "exemplarCity";
        "Makassar";
    }
    "Manila";
    {
        "exemplarCity";
        "Manila";
    }
    "Muscat";
    {
        "exemplarCity";
        "Maskat";
    }
}
```

```
"Nicosia";
{
  "exemplarCity";
  "Nikosia";
}
"Novokuznetsk";
{
  "exemplarCity";
  "Nowokuznetsk";
}
"Novosibirsk";
{
  "exemplarCity";
  "Nowosibirsk";
}
"Omsk";
{
  "exemplarCity";
  "Omsk";
}
"Oral";
{
  "exemplarCity";
  "Oral";
}
"Phnom_Penh";
{
  "exemplarCity";
  "Phnom Penh";
}
"Pontianak";
{
  "exemplarCity";
  "Pontianak";
}
"Pyongyang";
{
  "exemplarCity";
  "Pjöngjang";
}
"Qatar";
{
  "exemplarCity";
  "Katar";
}
"Qyzylorda";
{
  "exemplarCity";
  "Qysylorda";
}
"Rangoon";
{
  "exemplarCity";
  "Rangun";
}
"Riyadh";
{
```

```
        "exemplarCity";
        "Riad";
    }
    "Saigon";
    {
        "exemplarCity";
        "Ho-Chi-Minh-Stadt";
    }
    "Sakhalin";
    {
        "exemplarCity";
        "Sachalin";
    }
    "Samarkand";
    {
        "exemplarCity";
        "Samarkand";
    }
    "Seoul";
    {
        "exemplarCity";
        "Seoul";
    }
    "Shanghai";
    {
        "exemplarCity";
        "Shanghai";
    }
    "Singapore";
    {
        "exemplarCity";
        "Singapur";
    }
    "Srednekolymsk";
    {
        "exemplarCity";
        "Srednekolymsk";
    }
    "Taipei";
    {
        "exemplarCity";
        "Taipeh";
    }
    "Tashkent";
    {
        "exemplarCity";
        "Taschkent";
    }
    "Tbilisi";
    {
        "exemplarCity";
        "Tiflis";
    }
    "Tehran";
    {
        "exemplarCity";
        "Teheran";
    }
```

```
}
  "Thimphu";
  {
    "exemplarCity";
    "Thimphu";
  }
  "Tokyo";
  {
    "exemplarCity";
    "Tokio";
  }
  "Tomsk";
  {
    "exemplarCity";
    "Tomsk";
  }
  "Ulaanbaatar";
  {
    "exemplarCity";
    "Ulaanbaatar";
  }
  "Urumqi";
  {
    "exemplarCity";
    "Ürümqi";
  }
  "Ust-Nera";
  {
    "exemplarCity";
    "Ust-Nera";
  }
  "Vientiane";
  {
    "exemplarCity";
    "Vientiane";
  }
  "Vladivostok";
  {
    "exemplarCity";
    "Wladiwostok";
  }
  "Yakutsk";
  {
    "exemplarCity";
    "Jakutsk";
  }
  "Yekaterinburg";
  {
    "exemplarCity";
    "Jekaterinburg";
  }
  "Yerevan";
  {
    "exemplarCity";
    "Eriwan";
  }
}
```

```
"Indian";
{
  "Antananarivo";
  {
    "exemplarCity";
    "Antananarivo";
  }
  "Chagos";
  {
    "exemplarCity";
    "Chagos";
  }
  "Christmas";
  {
    "exemplarCity";
    "Weihnachtsinsel";
  }
  "Cocos";
  {
    "exemplarCity";
    "Cocos";
  }
  "Comoro";
  {
    "exemplarCity";
    "Komoren";
  }
  "Kerguelen";
  {
    "exemplarCity";
    "Kerguelen";
  }
  "Mahe";
  {
    "exemplarCity";
    "Mahe";
  }
  "Maldives";
  {
    "exemplarCity";
    "Malediven";
  }
  "Mauritius";
  {
    "exemplarCity";
    "Mauritius";
  }
  "Mayotte";
  {
    "exemplarCity";
    "Mayotte";
  }
  "Reunion";
  {
    "exemplarCity";
    "Réunion";
  }
}
```

```
}
"Australia";
{
  "Adelaide";
  {
    "exemplarCity";
    "Adelaide";
  }
  "Brisbane";
  {
    "exemplarCity";
    "Brisbane";
  }
  "Broken_Hill";
  {
    "exemplarCity";
    "Broken Hill";
  }
  "Currie";
  {
    "exemplarCity";
    "Currie";
  }
  "Darwin";
  {
    "exemplarCity";
    "Darwin";
  }
  "Eucla";
  {
    "exemplarCity";
    "Eucla";
  }
  "Hobart";
  {
    "exemplarCity";
    "Hobart";
  }
  "Lindeman";
  {
    "exemplarCity";
    "Lindeman";
  }
  "Lord_Howe";
  {
    "exemplarCity";
    "Lord Howe";
  }
  "Melbourne";
  {
    "exemplarCity";
    "Melbourne";
  }
  "Perth";
  {
    "exemplarCity";
    "Perth";
  }
}
```

```
}
  "Sydney";
  {
    "exemplarCity";
    "Sydney";
  }
}
"Pacific";
{
  "Apia";
  {
    "exemplarCity";
    "Apia";
  }
  "Auckland";
  {
    "exemplarCity";
    "Auckland";
  }
  "Bougainville";
  {
    "exemplarCity";
    "Bougainville";
  }
  "Chatham";
  {
    "exemplarCity";
    "Chatham";
  }
  "Easter";
  {
    "exemplarCity";
    "Osterinsel";
  }
  "Efate";
  {
    "exemplarCity";
    "Efate";
  }
  "Enderbury";
  {
    "exemplarCity";
    "Enderbury";
  }
  "Fakaofu";
  {
    "exemplarCity";
    "Fakaofu";
  }
  "Fiji";
  {
    "exemplarCity";
    "Fidschi";
  }
  "Funafuti";
  {
    "exemplarCity";
```

```
        "Funafuti";
    }
    "Galapagos";
    {
        "exemplarCity";
        "Galapagos";
    }
    "Gambier";
    {
        "exemplarCity";
        "Gambier";
    }
    "Guadalcanal";
    {
        "exemplarCity";
        "Guadalcanal";
    }
    "Guam";
    {
        "exemplarCity";
        "Guam";
    }
    "Honolulu";
    {
        "exemplarCity";
        "Honolulu";
    }
    "Johnston";
    {
        "exemplarCity";
        "Johnston";
    }
    "Kiritimati";
    {
        "exemplarCity";
        "Kiritimati";
    }
    "Kosrae";
    {
        "exemplarCity";
        "Kosrae";
    }
    "Kwajalein";
    {
        "exemplarCity";
        "Kwajalein";
    }
    "Majuro";
    {
        "exemplarCity";
        "Majuro";
    }
    "Marquesas";
    {
        "exemplarCity";
        "Marquesas";
    }
}
```



```
"Midway";
{
  "exemplarCity";
  "Midway";
}
"Nauru";
{
  "exemplarCity";
  "Nauru";
}
"Niue";
{
  "exemplarCity";
  "Niue";
}
"Norfolk";
{
  "exemplarCity";
  "Norfolk";
}
"Noumea";
{
  "exemplarCity";
  "Noumea";
}
"Pago_Pago";
{
  "exemplarCity";
  "Pago Pago";
}
"Palau";
{
  "exemplarCity";
  "Palau";
}
"Pitcairn";
{
  "exemplarCity";
  "Pitcairn";
}
"Ponape";
{
  "exemplarCity";
  "Pohnpei";
}
"Port_Moresby";
{
  "exemplarCity";
  "Port Moresby";
}
"Rarotonga";
{
  "exemplarCity";
  "Rarotonga";
}
"Saipan";
{
```

```

        "exemplarCity";
        "Saipan";
    }
    "Tahiti";
    {
        "exemplarCity";
        "Tahiti";
    }
    "Tarawa";
    {
        "exemplarCity";
        "Tarawa";
    }
    "Tongatapu";
    {
        "exemplarCity";
        "Tongatapu";
    }
    "Truk";
    {
        "exemplarCity";
        "Chuuk";
    }
    "Wake";
    {
        "exemplarCity";
        "Wake";
    }
    "Wallis";
    {
        "exemplarCity";
        "Wallis";
    }
}
"Arctic";
{
    "Longyearbyen";
    {
        "exemplarCity";
        "Longyearbyen";
    }
}
"Antarctica";
{
    "Casey";
    {
        "exemplarCity";
        "Casey";
    }
    "Davis";
    {
        "exemplarCity";
        "Davis";
    }
    "DumontDUrville";
    {
        "exemplarCity";

```

```

        "Dumont d'Urville";
    }
    "Macquarie";
    {
        "exemplarCity";
        "Macquarie";
    }
    "Mawson";
    {
        "exemplarCity";
        "Mawson";
    }
    "McMurdo";
    {
        "exemplarCity";
        "McMurdo";
    }
    "Palmer";
    {
        "exemplarCity";
        "Palmer";
    }
    "Rothera";
    {
        "exemplarCity";
        "Rothera";
    }
    "Syowa";
    {
        "exemplarCity";
        "Syowa";
    }
    "Troll";
    {
        "exemplarCity";
        "Troll";
    }
    "Vostok";
    {
        "exemplarCity";
        "Wostok";
    }
}
"Etc";
{
    "GMT";
    {
        "exemplarCity";
        "GMT";
    }
    "GMT1";
    {
        "exemplarCity";
        "GMT+1";
    }
    "GMT10";
    {

```

```
        "exemplarCity";
        "GMT+10";
    }
    "GMT11";
    {
        "exemplarCity";
        "GMT+11";
    }
    "GMT12";
    {
        "exemplarCity";
        "GMT+12";
    }
    "GMT2";
    {
        "exemplarCity";
        "GMT+2";
    }
    "GMT3";
    {
        "exemplarCity";
        "GMT+3";
    }
    "GMT4";
    {
        "exemplarCity";
        "GMT+4";
    }
    "GMT5";
    {
        "exemplarCity";
        "GMT+5";
    }
    "GMT6";
    {
        "exemplarCity";
        "GMT+6";
    }
    "GMT7";
    {
        "exemplarCity";
        "GMT+7";
    }
    "GMT8";
    {
        "exemplarCity";
        "GMT+8";
    }
    "GMT9";
    {
        "exemplarCity";
        "GMT+9";
    }
    "GMT-1";
    {
        "exemplarCity";
        "GMT-1";
    }
```

```
}
"GMT-10";
{
  "exemplarCity";
  "GMT-10";
}
"GMT-11";
{
  "exemplarCity";
  "GMT-11";
}
"GMT-12";
{
  "exemplarCity";
  "GMT-12";
}
"GMT-13";
{
  "exemplarCity";
  "GMT-13";
}
"GMT-14";
{
  "exemplarCity";
  "GMT-14";
}
"GMT-2";
{
  "exemplarCity";
  "GMT-2";
}
"GMT-3";
{
  "exemplarCity";
  "GMT-3";
}
"GMT-4";
{
  "exemplarCity";
  "GMT-4";
}
"GMT-5";
{
  "exemplarCity";
  "GMT-5";
}
"GMT-6";
{
  "exemplarCity";
  "GMT-6";
}
"GMT-7";
{
  "exemplarCity";
  "GMT-7";
}
"GMT-8";
```

```

        {
            "exemplarCity";
            "GMT-8";
        }
        "GMT-9";
        {
            "exemplarCity";
            "GMT-9";
        }
        "Unknown";
        {
            "exemplarCity";
            "Unbekannt";
        }
    }
}
"metazone";
{
    "Acre";
    {
        "long";
        {
            "generic";
            "Acre-Zeit",
            "standard";
            "Acre-Normalzeit",
            "daylight";
            "Acre-Sommerzeit";
        }
    }
    "Afghanistan";
    {
        "long";
        {
            "standard";
            "Afghanistan-Zeit";
        }
    }
    "Africa_Central";
    {
        "long";
        {
            "standard";
            "Zentralafrikanische Zeit";
        }
    }
    "Africa_Eastern";
    {
        "long";
        {
            "standard";
            "Ostafrikanische Zeit";
        }
    }
    "Africa_Southern";
    {
        "long";
    }
}

```

```

        {
            "standard";
            "Südafrikanische Zeit";
        }
    }
    "Africa_Western";
    {
        "long";
        {
            "generic";
            "Westafrikanische Zeit",
            "standard";
            "Westafrikanische Normalzeit",
            "daylight";
            "Westafrikanische Sommerzeit";
        }
    }
    "Alaska";
    {
        "long";
        {
            "generic";
            "Alaska-Zeit",
            "standard";
            "Alaska-Normalzeit",
            "daylight";
            "Alaska-Sommerzeit";
        }
    }
    "Almaty";
    {
        "long";
        {
            "generic";
            "Almaty-Zeit",
            "standard";
            "Almaty-Normalzeit",
            "daylight";
            "Almaty-Sommerzeit";
        }
    }
    "Amazon";
    {
        "long";
        {
            "generic";
            "Amazonas-Zeit",
            "standard";
            "Amazonas-Normalzeit",
            "daylight";
            "Amazonas-Sommerzeit";
        }
    }
    "America_Central";
    {
        "long";
        {

```

```

        "generic";
        "Nordamerikanische Inlandzeit",
        "standard";
        "Nordamerikanische Inland-Normalzeit",
        "daylight";
        "Nordamerikanische Inland-Sommerzeit";
    }
}
"America_Eastern";
{
    "long";
    {
        "generic";
        "Nordamerikanische Ostküstenzeit",
        "standard";
        "Nordamerikanische Ostküsten-Normalzeit",
        "daylight";
        "Nordamerikanische Ostküsten-Sommerzeit";
    }
}
"America_Mountain";
{
    "long";
    {
        "generic";
        "Rocky-Mountain-Zeit",
        "standard";
        "Rocky Mountain-Normalzeit",
        "daylight";
        "Rocky-Mountain-Sommerzeit";
    }
}
"America_Pacific";
{
    "long";
    {
        "generic";
        "Nordamerikanische Westküstenzeit",
        "standard";
        "Nordamerikanische Westküsten-Normalzeit",
        "daylight";
        "Nordamerikanische Westküsten-Sommerzeit";
    }
}
"Anadyr";
{
    "long";
    {
        "generic";
        "Anadyr Zeit",
        "standard";
        "Anadyr Normalzeit",
        "daylight";
        "Anadyr Sommerzeit";
    }
}
"Apia";

```



```

    {
      "long";
      {
        "generic";
        "Apia-Zeit",
        "standard";
        "Apia-Normalzeit",
        "daylight";
        "Apia-Sommerzeit";
      }
    }
    "Aqtau";
    {
      "long";
      {
        "generic";
        "Aqtau-Zeit",
        "standard";
        "Aqtau-Normalzeit",
        "daylight";
        "Aqtau-Sommerzeit";
      }
    }
    "Aqtobe";
    {
      "long";
      {
        "generic";
        "Aqtöbe-Zeit",
        "standard";
        "Aqtöbe-Normalzeit",
        "daylight";
        "Aqtöbe-Sommerzeit";
      }
    }
    "Arabian";
    {
      "long";
      {
        "generic";
        "Arabische Zeit",
        "standard";
        "Arabische Normalzeit",
        "daylight";
        "Arabische Sommerzeit";
      }
    }
    "Argentina";
    {
      "long";
      {
        "generic";
        "Argentinische Zeit",
        "standard";
        "Argentinische Normalzeit",
        "daylight";
        "Argentinische Sommerzeit";
      }
    }
  }

```

```

    }
  }
  "Argentina_Western";
  {
    "long";
    {
      "generic";
      "Westargentinische Zeit",
      "standard";
      "Westargentinische Normalzeit",
      "daylight";
      "Westargentinische Sommerzeit";
    }
  }
  "Armenia";
  {
    "long";
    {
      "generic";
      "Armenische Zeit",
      "standard";
      "Armenische Normalzeit",
      "daylight";
      "Armenische Sommerzeit";
    }
  }
  "Atlantic";
  {
    "long";
    {
      "generic";
      "Atlantik-Zeit",
      "standard";
      "Atlantik-Normalzeit",
      "daylight";
      "Atlantik-Sommerzeit";
    }
  }
  "Australia_Central";
  {
    "long";
    {
      "generic";
      "Zentralaustralische Zeit",
      "standard";
      "Zentralaustralische Normalzeit",
      "daylight";
      "Zentralaustralische Sommerzeit";
    }
  }
  "Australia_CentralWestern";
  {
    "long";
    {
      "generic";
      "Zentral-/Westaustralische Zeit",
      "standard";

```

```

        "Zentral-/Westaustralische Normalzeit",
        "daylight";
        "Zentral-/Westaustralische Sommerzeit";
    }
}
"Australia_Eastern";
{
    "long";
    {
        "generic";
        "Ostaustralische Zeit",
        "standard";
        "Ostaustralische Normalzeit",
        "daylight";
        "Ostaustralische Sommerzeit";
    }
}
"Australia_Western";
{
    "long";
    {
        "generic";
        "Westaustralische Zeit",
        "standard";
        "Westaustralische Normalzeit",
        "daylight";
        "Westaustralische Sommerzeit";
    }
}
"Azerbaijan";
{
    "long";
    {
        "generic";
        "Aserbajdschanische Zeit",
        "standard";
        "Aserbajdschanische Normalzeit",
        "daylight";
        "Aserbajdschanische Sommerzeit";
    }
}
"Azores";
{
    "long";
    {
        "generic";
        "Azoren-Zeit",
        "standard";
        "Azoren-Normalzeit",
        "daylight";
        "Azoren-Sommerzeit";
    }
}
"Bangladesh";
{
    "long";
    {

```

```

        "generic";
        "Bangladesch-Zeit",
        "standard";
        "Bangladesch-Normalzeit",
        "daylight";
        "Bangladesch-Sommerzeit";
    }
}
"Bhutan";
{
    "long";
    {
        "standard";
        "Bhutan-Zeit";
    }
}
"Bolivia";
{
    "long";
    {
        "standard";
        "Bolivianische Zeit";
    }
}
"Brasilia";
{
    "long";
    {
        "generic";
        "Brasília-Zeit",
        "standard";
        "Brasília-Normalzeit",
        "daylight";
        "Brasília-Sommerzeit";
    }
}
"Brunei";
{
    "long";
    {
        "standard";
        "Brunei-Zeit";
    }
}
"Cape_Verde";
{
    "long";
    {
        "generic";
        "Cabo-Verde-Zeit",
        "standard";
        "Cabo-Verde-Normalzeit",
        "daylight";
        "Cabo-Verde-Sommerzeit";
    }
}
"Casey";

```

```

    {
      "long";
      {
        "standard";
        "Casey-Zeit";
      }
    }
    "Chamorro";
    {
      "long";
      {
        "standard";
        "Chamorro-Zeit";
      }
    }
    "Chatham";
    {
      "long";
      {
        "generic";
        "Chatham-Zeit",
        "standard";
        "Chatham-Normalzeit",
        "daylight";
        "Chatham-Sommerzeit";
      }
    }
    "Chile";
    {
      "long";
      {
        "generic";
        "Chilenische Zeit",
        "standard";
        "Chilenische Normalzeit",
        "daylight";
        "Chilenische Sommerzeit";
      }
    }
    "China";
    {
      "long";
      {
        "generic";
        "Chinesische Zeit",
        "standard";
        "Chinesische Normalzeit",
        "daylight";
        "Chinesische Sommerzeit";
      }
    }
    "Choibalsan";
    {
      "long";
      {
        "generic";
        "Tschoibalsan-Zeit",

```

```

        "standard";
        "Tschoibalsan-Normalzeit",
        "daylight";
        "Tschoibalsan-Sommerzeit";
    }
}
"Christmas";
{
    "long";
    {
        "standard";
        "Weihnachtsinsel-Zeit";
    }
}
"Cocos";
{
    "long";
    {
        "standard";
        "Kokosinseln-Zeit";
    }
}
"Colombia";
{
    "long";
    {
        "generic";
        "Kolumbianische Zeit",
        "standard";
        "Kolumbianische Normalzeit",
        "daylight";
        "Kolumbianische Sommerzeit";
    }
}
"Cook";
{
    "long";
    {
        "generic";
        "Cookinseln-Zeit",
        "standard";
        "Cookinseln-Normalzeit",
        "daylight";
        "Cookinseln-Sommerzeit";
    }
}
"Cuba";
{
    "long";
    {
        "generic";
        "Kubanische Zeit",
        "standard";
        "Kubanische Normalzeit",
        "daylight";
        "Kubanische Sommerzeit";
    }
}

```

```

    }
    "Davis";
    {
        "long";
        {
            "standard";
            "Davis-Zeit";
        }
    }
    "DumontDUrville";
    {
        "long";
        {
            "standard";
            "Dumont-d'Urville-Zeit";
        }
    }
    "East_Timor";
    {
        "long";
        {
            "standard";
            "Osttimor-Zeit";
        }
    }
    "Easter";
    {
        "long";
        {
            "generic";
            "Osterinsel-Zeit",
            "standard";
            "Osterinsel-Normalzeit",
            "daylight";
            "Osterinsel-Sommerzeit";
        }
    }
    "Ecuador";
    {
        "long";
        {
            "standard";
            "Ecuadorianische Zeit";
        }
    }
    "Europe_Central";
    {
        "long";
        {
            "generic";
            "Mittleuropäische Zeit",
            "standard";
            "Mittleuropäische Normalzeit",
            "daylight";
            "Mittleuropäische Sommerzeit";
        }
        "short";
    }

```

```

        {
            "generic";
            "MEZ",
            "standard";
            "MEZ",
            "daylight";
            "MESZ";
        }
    }
    "Europe_Eastern";
    {
        "long";
        {
            "generic";
            "Osteuropäische Zeit",
            "standard";
            "Osteuropäische Normalzeit",
            "daylight";
            "Osteuropäische Sommerzeit";
        }
        "short";
        {
            "generic";
            "OEZ",
            "standard";
            "OEZ",
            "daylight";
            "OESZ";
        }
    }
    "Europe_Further_Eastern";
    {
        "long";
        {
            "standard";
            "Kaliningrader Zeit";
        }
    }
    "Europe_Western";
    {
        "long";
        {
            "generic";
            "Westeuropäische Zeit",
            "standard";
            "Westeuropäische Normalzeit",
            "daylight";
            "Westeuropäische Sommerzeit";
        }
        "short";
        {
            "generic";
            "WEZ",
            "standard";
            "WEZ",
            "daylight";
            "WESZ";
        }
    }

```



```

    }
  }
  "Falkland";
  {
    "long";
    {
      "generic";
      "Falklandinseln-Zeit",
      "standard";
      "Falklandinseln-Normalzeit",
      "daylight";
      "Falklandinseln-Sommerzeit";
    }
  }
  "Fiji";
  {
    "long";
    {
      "generic";
      "Fidschi-Zeit",
      "standard";
      "Fidschi-Normalzeit",
      "daylight";
      "Fidschi-Sommerzeit";
    }
  }
  "French_Guiana";
  {
    "long";
    {
      "standard";
      "Französisch-Guayana-Zeit";
    }
  }
  "French_Southern";
  {
    "long";
    {
      "standard";
      "Französische Süd- und Antarktisgebiete-
Zeit";
    }
  }
  "Galapagos";
  {
    "long";
    {
      "standard";
      "Galapagos-Zeit";
    }
  }
  "Gambier";
  {
    "long";
    {
      "standard";
      "Gambier-Zeit";
    }
  }

```

```

    }
  }
  "Georgia";
  {
    "long";
    {
      "generic";
      "Georgische Zeit",
      "standard";
      "Georgische Normalzeit",
      "daylight";
      "Georgische Sommerzeit";
    }
  }
  "Gilbert_Islands";
  {
    "long";
    {
      "standard";
      "Gilbert-Inseln-Zeit";
    }
  }
  "GMT";
  {
    "long";
    {
      "standard";
      "Mittlere Greenwich-Zeit";
    }
  }
  "Greenland_Eastern";
  {
    "long";
    {
      "generic";
      "Ostgrönland-Zeit",
      "standard";
      "Ostgrönland-Normalzeit",
      "daylight";
      "Ostgrönland-Sommerzeit";
    }
  }
  "Greenland_Western";
  {
    "long";
    {
      "generic";
      "Westgrönland-Zeit",
      "standard";
      "Westgrönland-Normalzeit",
      "daylight";
      "Westgrönland-Sommerzeit";
    }
  }
  "Guam";
  {
    "long";

```

```

        {
            "standard";
            "Guam-Zeit";
        }
    }
    "Gulf";
    {
        "long";
        {
            "standard";
            "Golf-Zeit";
        }
    }
    "Guyana";
    {
        "long";
        {
            "standard";
            "Guyana-Zeit";
        }
    }
    "Hawaii_Aleutian";
    {
        "long";
        {
            "generic";
            "Hawaii-Aleuten-Zeit",
            "standard";
            "Hawaii-Aleuten-Normalzeit",
            "daylight";
            "Hawaii-Aleuten-Sommerzeit";
        }
    }
    "Hong_Kong";
    {
        "long";
        {
            "generic";
            "Hongkong-Zeit",
            "standard";
            "Hongkong-Normalzeit",
            "daylight";
            "Hongkong-Sommerzeit";
        }
    }
    "Hovd";
    {
        "long";
        {
            "generic";
            "Chowd-Zeit",
            "standard";
            "Chowd-Normalzeit",
            "daylight";
            "Chowd-Sommerzeit";
        }
    }
}

```

```
"India";
{
  "long";
  {
    "standard";
    "Indische Zeit";
  }
}
"Indian_Ocean";
{
  "long";
  {
    "standard";
    "Indischer Ozean-Zeit";
  }
}
"Indochina";
{
  "long";
  {
    "standard";
    "Indochina-Zeit";
  }
}
"Indonesia_Central";
{
  "long";
  {
    "standard";
    "Zentralindonesische Zeit";
  }
}
"Indonesia_Eastern";
{
  "long";
  {
    "standard";
    "Ostindonesische Zeit";
  }
}
"Indonesia_Western";
{
  "long";
  {
    "standard";
    "Westindonesische Zeit";
  }
}
"Iran";
{
  "long";
  {
    "generic";
    "Iranische Zeit",
    "standard";
    "Iranische Normalzeit",
    "daylight";
  }
}
```

```

        "Iranische Sommerzeit";
    }
}
"Irkutsk";
{
    "long";
    {
        "generic";
        "Irkutsk-Zeit",
        "standard";
        "Irkutsk-Normalzeit",
        "daylight";
        "Irkutsk-Sommerzeit";
    }
}
"Israel";
{
    "long";
    {
        "generic";
        "Israelische Zeit",
        "standard";
        "Israelische Normalzeit",
        "daylight";
        "Israelische Sommerzeit";
    }
}
"Japan";
{
    "long";
    {
        "generic";
        "Japanische Zeit",
        "standard";
        "Japanische Normalzeit",
        "daylight";
        "Japanische Sommerzeit";
    }
}
"Kamchatka";
{
    "long";
    {
        "generic";
        "Kamtschatka-Zeit",
        "standard";
        "Kamtschatka-Normalzeit",
        "daylight";
        "Kamtschatka-Sommerzeit";
    }
}
"Kazakhstan_Eastern";
{
    "long";
    {
        "standard";
        "Ostkasachische Zeit";
    }
}

```

```

    }
  }
  "Kazakhstan_Western";
  {
    "long";
    {
      "standard";
      "Westkasachische Zeit";
    }
  }
  "Korea";
  {
    "long";
    {
      "generic";
      "Koreanische Zeit",
      "standard";
      "Koreanische Normalzeit",
      "daylight";
      "Koreanische Sommerzeit";
    }
  }
  "Kosrae";
  {
    "long";
    {
      "standard";
      "Kosrae-Zeit";
    }
  }
  "Krasnoyarsk";
  {
    "long";
    {
      "generic";
      "Krasnojarsk-Zeit",
      "standard";
      "Krasnojarsk-Normalzeit",
      "daylight";
      "Krasnojarsk-Sommerzeit";
    }
  }
  "Kyrgystan";
  {
    "long";
    {
      "standard";
      "Kirgisistan-Zeit";
    }
  }
  "Lanka";
  {
    "long";
    {
      "standard";
      "Sri-Lanka-Zeit";
    }
  }

```

```

    }
    "Line_Islands";
    {
        "long";
        {
            "standard";
            "Linieninseln-Zeit";
        }
    }
    "Lord_Howe";
    {
        "long";
        {
            "generic";
            "Lord-Howe-Zeit",
            "standard";
            "Lord-Howe-Normalzeit",
            "daylight";
            "Lord-Howe-Sommerzeit";
        }
    }
    "Macau";
    {
        "long";
        {
            "generic";
            "Macau-Zeit",
            "standard";
            "Macau-Normalzeit",
            "daylight";
            "Macau-Sommerzeit";
        }
    }
    "Macquarie";
    {
        "long";
        {
            "standard";
            "Macquarieinsel-Zeit";
        }
    }
    "Magadan";
    {
        "long";
        {
            "generic";
            "Magadan-Zeit",
            "standard";
            "Magadan-Normalzeit",
            "daylight";
            "Magadan-Sommerzeit";
        }
    }
    "Malaysia";
    {
        "long";
        {

```

```

        "standard";
        "Malaysische Zeit";
    }
}
"Maldives";
{
    "long";
    {
        "standard";
        "Malediven-Zeit";
    }
}
"Marquesas";
{
    "long";
    {
        "standard";
        "Marquesas-Zeit";
    }
}
"Marshall_Islands";
{
    "long";
    {
        "standard";
        "Marshallinseln-Zeit";
    }
}
"Mauritius";
{
    "long";
    {
        "generic";
        "Mauritius-Zeit",
        "standard";
        "Mauritius-Normalzeit",
        "daylight";
        "Mauritius-Sommerzeit";
    }
}
"Mawson";
{
    "long";
    {
        "standard";
        "Mawson-Zeit";
    }
}
"Mexico_Northwest";
{
    "long";
    {
        "generic";
        "Mexiko Nordwestliche Zone-Zeit",
        "standard";
        "Mexiko Nordwestliche Zone-Normalzeit",
        "daylight";
    }
}

```



```
        "Mexiko Nordwestliche Zone-Sommerzeit";
    }
}
"Mexico_Pacific";
{
    "long";
    {
        "generic";
        "Mexiko Pazifikzone-Zeit",
        "standard";
        "Mexiko Pazifikzone-Normalzeit",
        "daylight";
        "Mexiko Pazifikzone-Sommerzeit";
    }
}
"Mongolia";
{
    "long";
    {
        "generic";
        "Ulaanbaatar-Zeit",
        "standard";
        "Ulaanbaatar-Normalzeit",
        "daylight";
        "Ulaanbaatar-Sommerzeit";
    }
}
"Moscow";
{
    "long";
    {
        "generic";
        "Moskauer Zeit",
        "standard";
        "Moskauer Normalzeit",
        "daylight";
        "Moskauer Sommerzeit";
    }
}
"Myanmar";
{
    "long";
    {
        "standard";
        "Myanmar-Zeit";
    }
}
"Nauru";
{
    "long";
    {
        "standard";
        "Nauru-Zeit";
    }
}
"Nepal";
{
```

```

        "long";
        {
            "standard";
            "Nepalesische Zeit";
        }
    }
    "New_Caledonia";
    {
        "long";
        {
            "generic";
            "Neukaledonische Zeit",
            "standard";
            "Neukaledonische Normalzeit",
            "daylight";
            "Neukaledonische Sommerzeit";
        }
    }
    "New_Zealand";
    {
        "long";
        {
            "generic";
            "Neuseeland-Zeit",
            "standard";
            "Neuseeland-Normalzeit",
            "daylight";
            "Neuseeland-Sommerzeit";
        }
    }
    "Newfoundland";
    {
        "long";
        {
            "generic";
            "Neufundland-Zeit",
            "standard";
            "Neufundland-Normalzeit",
            "daylight";
            "Neufundland-Sommerzeit";
        }
    }
    "Niue";
    {
        "long";
        {
            "standard";
            "Niue-Zeit";
        }
    }
    "Norfolk";
    {
        "long";
        {
            "standard";
            "Norfolkinsel-Zeit";
        }
    }

```

```

    }
    "Noronha";
    {
        "long";
        {
            "generic";
            "Fernando de Noronha-Zeit",
            "standard";
            "Fernando de Noronha-Normalzeit",
            "daylight";
            "Fernando de Noronha-Sommerzeit";
        }
    }
    "North_Mariana";
    {
        "long";
        {
            "standard";
            "Nördliche-Marianen-Zeit";
        }
    }
    "Novosibirsk";
    {
        "long";
        {
            "generic";
            "Nowosibirsk-Zeit",
            "standard";
            "Nowosibirsk-Normalzeit",
            "daylight";
            "Nowosibirsk-Sommerzeit";
        }
    }
    "Omsk";
    {
        "long";
        {
            "generic";
            "Omsk-Zeit",
            "standard";
            "Omsk-Normalzeit",
            "daylight";
            "Omsk-Sommerzeit";
        }
    }
    "Pakistan";
    {
        "long";
        {
            "generic";
            "Pakistanische Zeit",
            "standard";
            "Pakistanische Normalzeit",
            "daylight";
            "Pakistanische Sommerzeit";
        }
    }
}

```

```
"Palau";
{
  "long";
  {
    "standard";
    "Palau-Zeit";
  }
}
"Papua_New_Guinea";
{
  "long";
  {
    "standard";
    "Papua-Neuguinea-Zeit";
  }
}
"Paraguay";
{
  "long";
  {
    "generic";
    "Paraguayianische Zeit",
    "standard";
    "Paraguayianische Normalzeit",
    "daylight";
    "Paraguayianische Sommerzeit";
  }
}
"Peru";
{
  "long";
  {
    "generic";
    "Peruanische Zeit",
    "standard";
    "Peruanische Normalzeit",
    "daylight";
    "Peruanische Sommerzeit";
  }
}
"Philippines";
{
  "long";
  {
    "generic";
    "Philippinische Zeit",
    "standard";
    "Philippinische Normalzeit",
    "daylight";
    "Philippinische Sommerzeit";
  }
}
"Phoenix_Islands";
{
  "long";
  {
    "standard";
```

```

        "Phoenixinseln-Zeit";
    }
}
"Pierre_Miquelon";
{
    "long";
    {
        "generic";
        "Saint-Pierre-und-Miquelon-Zeit",
        "standard";
        "Saint-Pierre-und-Miquelon-Normalzeit",
        "daylight";
        "Saint-Pierre-und-Miquelon-Sommerzeit";
    }
}
"Pitcairn";
{
    "long";
    {
        "standard";
        "Pitcairninseln-Zeit";
    }
}
"Ponape";
{
    "long";
    {
        "standard";
        "Ponape-Zeit";
    }
}
"Pyongyang";
{
    "long";
    {
        "standard";
        "Pjöngjang-Zeit";
    }
}
"Qyzylorda";
{
    "long";
    {
        "generic";
        "Quysylorda-Zeit",
        "standard";
        "Quysylorda-Normalzeit",
        "daylight";
        "Qysylorda-Sommerzeit";
    }
}
"Reunion";
{
    "long";
    {
        "standard";
        "Réunion-Zeit";
    }
}

```

```

    }
  }
  "Rothera";
  {
    "long";
    {
      "standard";
      "Rothera-Zeit";
    }
  }
  "Sakhalin";
  {
    "long";
    {
      "generic";
      "Sachalin-Zeit",
      "standard";
      "Sachalin-Normalzeit",
      "daylight";
      "Sachalin-Sommerzeit";
    }
  }
  "Samara";
  {
    "long";
    {
      "generic";
      "Samara-Zeit",
      "standard";
      "Samara-Normalzeit",
      "daylight";
      "Samara-Sommerzeit";
    }
  }
  "Samoa";
  {
    "long";
    {
      "generic";
      "Samoa-Zeit",
      "standard";
      "Samoa-Normalzeit",
      "daylight";
      "Samoa-Sommerzeit";
    }
  }
  "Seychelles";
  {
    "long";
    {
      "standard";
      "Seychellen-Zeit";
    }
  }
  "Singapore";
  {
    "long";

```

```

        {
            "standard";
            "Singapur-Zeit";
        }
    }
    "Solomon";
    {
        "long";
        {
            "standard";
            "Salomoninseln-Zeit";
        }
    }
    "South_Georgia";
    {
        "long";
        {
            "standard";
            "Südgeorgische Zeit";
        }
    }
    "Suriname";
    {
        "long";
        {
            "standard";
            "Suriname-Zeit";
        }
    }
    "Syowa";
    {
        "long";
        {
            "standard";
            "Syowa-Zeit";
        }
    }
    "Tahiti";
    {
        "long";
        {
            "standard";
            "Tahiti-Zeit";
        }
    }
    "Taipei";
    {
        "long";
        {
            "generic";
            "Taipeh-Zeit",
            "standard";
            "Taipeh-Normalzeit",
            "daylight";
            "Taipeh-Sommerzeit";
        }
    }
}

```

```
"Tajikistan";
{
  "long";
  {
    "standard";
    "Tadschikistan-Zeit";
  }
}
"Tokelau";
{
  "long";
  {
    "standard";
    "Tokelau-Zeit";
  }
}
"Tonga";
{
  "long";
  {
    "generic";
    "Tonganische Zeit",
    "standard";
    "Tonganische Normalzeit",
    "daylight";
    "Tonganische Sommerzeit";
  }
}
"Truk";
{
  "long";
  {
    "standard";
    "Chuuk-Zeit";
  }
}
"Turkmenistan";
{
  "long";
  {
    "generic";
    "Turkmenistan-Zeit",
    "standard";
    "Turkmenistan-Normalzeit",
    "daylight";
    "Turkmenistan-Sommerzeit";
  }
}
"Tuvalu";
{
  "long";
  {
    "standard";
    "Tuvalu-Zeit";
  }
}
"Uruguay";
```



```
{
  "long";
  {
    "generic";
    "Uruguayische Zeit",
    "standard";
    "Uruguayische Normalzeit",
    "daylight";
    "Uruguayische Sommerzeit";
  }
}
"Uzbekistan";
{
  "long";
  {
    "generic";
    "Usbekistan-Zeit",
    "standard";
    "Usbekistan-Normalzeit",
    "daylight";
    "Usbekistan-Sommerzeit";
  }
}
"Vanuatu";
{
  "long";
  {
    "generic";
    "Vanuatu-Zeit",
    "standard";
    "Vanuatu-Normalzeit",
    "daylight";
    "Vanuatu-Sommerzeit";
  }
}
"Venezuela";
{
  "long";
  {
    "standard";
    "Venezuela-Zeit";
  }
}
"Vladivostok";
{
  "long";
  {
    "generic";
    "Wladiwostok-Zeit",
    "standard";
    "Wladiwostok-Normalzeit",
    "daylight";
    "Wladiwostok-Sommerzeit";
  }
}
"Volgograd";
{
```

```
        "long";
        {
            "generic";
            "Wolgograd-Zeit",
            "standard";
            "Wolgograd-Normalzeit",
            "daylight";
            "Wolgograd-Sommerzeit";
        }
    }
    "Vostok";
    {
        "long";
        {
            "standard";
            "Wostok-Zeit";
        }
    }
    "Wake";
    {
        "long";
        {
            "standard";
            "Wake-Insel-Zeit";
        }
    }
    "Wallis";
    {
        "long";
        {
            "standard";
            "Wallis-und-Futuna-Zeit";
        }
    }
    "Yakutsk";
    {
        "long";
        {
            "generic";
            "Jakutsk-Zeit",
            "standard";
            "Jakutsk-Normalzeit",
            "daylight";
            "Jakutsk-Sommerzeit";
        }
    }
    "Yekaterinburg";
    {
        "long";
        {
            "generic";
            "Jekaterinburg-Zeit",
            "standard";
            "Jekaterinburg-Normalzeit",
            "daylight";
            "Jekaterinburg-Sommerzeit";
        }
    }
}
```



```

        "9": "س",
        "10": "ك",
        "11": "ب",
        "12": "د"
    },
    "wide": {
        "1": "يناير",
        "2": "فبراير",
        "3": "مارس",
        "4": "أبريل",
        "5": "مايو",
        "6": "يونيو",
        "7": "يوليو",
        "8": "أغسطس",
        "9": "سبتمبر",
        "10": "أكتوبر",
        "11": "نوفمبر",
        "12": "ديسمبر"
    }
},
"stand-alone": {
    "abbreviated": {
        "1": "يناير",
        "2": "فبراير",
        "3": "مارس",
        "4": "أبريل",
        "5": "مايو",
        "6": "يونيو",
        "7": "يوليو",
        "8": "أغسطس",
        "9": "سبتمبر",
        "10": "أكتوبر",
        "11": "نوفمبر",
        "12": "ديسمبر"
    },
    "narrow": {
        "1": "ي",
        "2": "ف",
        "3": "م",
        "4": "أ",
        "5": "و",
        "6": "ن",
        "7": "ل",
        "8": "غ",
        "9": "س",
        "10": "ك",
        "11": "ب",
        "12": "د"
    },
    "wide": {
        "1": "يناير",
        "2": "فبراير",
        "3": "مارس",
        "4": "أبريل",
        "5": "مايو",
        "6": "يونيو",
        "7": "يوليو",

```

```

        "8": "أغسطس",
        "9": "سبتمبر",
        "10": "أكتوبر",
        "11": "نوفمبر",
        "12": "ديسمبر"
    }
}
},
"days": {
    "format": {
        "abbreviated": {
            "sun": "الأحد",
            "mon": "الاثنين",
            "tue": "الثلاثاء",
            "wed": "الأربعاء",
            "thu": "الخميس",
            "fri": "الجمعة",
            "sat": "السبت"
        },
        "narrow": {
            "sun": "ح",
            "mon": "ن",
            "tue": "ث",
            "wed": "ر",
            "thu": "خ",
            "fri": "ج",
            "sat": "س"
        },
        "short": {
            "sun": "الأحد",
            "mon": "الاثنين",
            "tue": "الثلاثاء",
            "wed": "الأربعاء",
            "thu": "الخميس",
            "fri": "الجمعة",
            "sat": "السبت"
        },
        "wide": {
            "sun": "الأحد",
            "mon": "الاثنين",
            "tue": "الثلاثاء",
            "wed": "الأربعاء",
            "thu": "الخميس",
            "fri": "الجمعة",
            "sat": "السبت"
        }
    },
    "stand-alone": {
        "abbreviated": {
            "sun": "الأحد",
            "mon": "الاثنين",
            "tue": "الثلاثاء",
            "wed": "الأربعاء",
            "thu": "الخميس",
            "fri": "الجمعة",
            "sat": "السبت"
        }
    }
},

```

```

        "narrow": {
            "sun": "ح",
            "mon": "ن",
            "tue": "ث",
            "wed": "ر",
            "thu": "خ",
            "fri": "ج",
            "sat": "س"
        },
        "short": {
            "sun": "الأحد",
            "mon": "الاثنين",
            "tue": "الثلاثاء",
            "wed": "الأربعاء",
            "thu": "الخميس",
            "fri": "الجمعة",
            "sat": "السبت"
        },
        "wide": {
            "sun": "الأحد",
            "mon": "الاثنين",
            "tue": "الثلاثاء",
            "wed": "الأربعاء",
            "thu": "الخميس",
            "fri": "الجمعة",
            "sat": "السبت"
        }
    },
    "quarters": {
        "format": {
            "abbreviated": {
                "1": "الربع الأول",
                "2": "الربع الثاني",
                "3": "الربع الثالث",
                "4": "الربع الرابع"
            },
            "narrow": {
                "1": "١",
                "2": "٢",
                "3": "٣",
                "4": "٤"
            },
            "wide": {
                "1": "الربع الأول",
                "2": "الربع الثاني",
                "3": "الربع الثالث",
                "4": "الربع الرابع"
            }
        },
        "stand-alone": {
            "abbreviated": {
                "1": "الربع الأول",
                "2": "الربع الثاني",
                "3": "الربع الثالث",
                "4": "الربع الرابع"
            }
        }
    },

```

```

        "narrow": {
            "1": "١",
            "2": "٢",
            "3": "٣",
            "4": "٤"
        },
        "wide": {
            "1": "الربع الأول",
            "2": "الربع الثاني",
            "3": "الربع الثالث",
            "4": "الربع الرابع"
        }
    },
    "dayPeriods": {
        "format": {
            "abbreviated": {
                "am": "ص",
                "pm": "م",
                "morning1": "فجراً",
                "morning2": "ص",
                "afternoon1": "ظهراً",
                "afternoon2": "بعد الظهر",
                "evening1": "مساءً",
                "night1": "منتصف الليل",
                "night2": "ل"
            },
            "narrow": {
                "am": "ص",
                "pm": "م",
                "morning1": "فجراً",
                "morning2": "صباحاً",
                "afternoon1": "ظهراً",
                "afternoon2": "بعد الظهر",
                "evening1": "مساءً",
                "night1": "منتصف الليل",
                "night2": "ليلاً"
            },
            "wide": {
                "am": "ص",
                "pm": "م",
                "morning1": "فجراً",
                "morning2": "صباحاً",
                "afternoon1": "ظهراً",
                "afternoon2": "بعد الظهر",
                "evening1": "مساءً",
                "night1": "منتصف الليل",
                "night2": "ليلاً"
            }
        },
        "stand-alone": {
            "abbreviated": {
                "am": "ص",
                "pm": "م",
                "morning1": "فجراً",
                "morning2": "ص",
                "afternoon1": "ظهراً",

```

```

        "afternoon2": "بعد الظهر",
        "evening1": "مساءً",
        "night1": "منتصف الليل",
        "night2": "ليلاً"
    },
    "narrow": {
        "am": "ص",
        "pm": "م",
        "morning1": "فجراً",
        "morning2": "صباحاً",
        "afternoon1": "ظهراً",
        "afternoon2": "بعد الظهر",
        "evening1": "مساءً",
        "night1": "منتصف الليل",
        "night2": "ليلاً"
    },
    "wide": {
        "am": "صباحاً",
        "pm": "مساءً",
        "morning1": "فجراً",
        "morning2": "صباحاً",
        "afternoon1": "ظهراً",
        "afternoon2": "بعد الظهر",
        "evening1": "مساءً",
        "night1": "منتصف الليل",
        "night2": "ليلاً"
    }
    },
    "eras": {
        "eraNames": {
            "0": "قبل الميلاد",
            "0-alt-variant": "BCE",
            "1": "ميلادي",
            "1-alt-variant": "بعد الميلاد"
        },
        "eraAbbr": {
            "0": "ق.م",
            "0-alt-variant": "BCE",
            "1": "م",
            "1-alt-variant": "ب.م"
        },
        "eraNarrow": {
            "0": "ق.م",
            "0-alt-variant": "BCE",
            "1": "م",
            "1-alt-variant": "ب.م"
        }
    },
    "dateFormats": {
        "full": "EEEE، d MMMM، y",
        "long": "d MMMM، y",
        "medium": "dd/MM/y",
        "short": "d/M/y"
    },
    "timeFormats": {
        "full": "h:mm:ss a zzzz",

```



```

    "long": "h:mm:ss a z",
    "medium": "h:mm:ss a",
    "short": "h:mm a"
  },
  "dateTimeFormats": {
    "full": "{1} {0}",
    "long": "{1} {0}",
    "medium": "{1} {0}",
    "short": "{1} {0}",
    "availableFormats": {
      "d": "d",
      "E": "ccc",
      "Ed": "E, d",
      "Ehm": "E h:mm a",
      "EHm": "E HH:mm",
      "Ehms": "E h:mm:ss a",
      "EHms": "E HH:mm:ss",
      "Gy": "y G",
      "GyMMM": "MMM y G",
      "GyMMMd": "d MMM, y G",
      "GyMMMED": "E, d MMM, y G",
      "h": "h a",
      "H": "HH",
      "hm": "h:mm a",
      "Hm": "HH:mm",
      "hms": "h:mm:ss a",
      "Hms": "HH:mm:ss",
      "hmsv": "h:mm:ss a v",
      "Hmsv": "HH:mm:ss v",
      "hmv": "h:mm a v",
      "Hmv": "HH:mm v",
      "M": "L",
      "Md": "d/M",
      "MEd": "E, d/M",
      "MMdd": "dd/MM",
      "MMM": "LLL",
      "MMMd": "d MMM",
      "MMMED": "E, d MMM",
      "MMMMd": "d MMMM",
      "MMMMEd": "E, d MMMM",
      "MMMMMW": "الأسبوع W من MMM",
      "MMMMMW": "الأسبوع W من MMM",
      "MMMMMW": "الأسبوع W من MMM",
      "MMMMMW": "الأسبوع W من MMM",
      "MMMMMW": "الأسبوع W من MMM",
      "MMMMMW": "الأسبوع W من MMM",
      "ms": "mm:ss",
      "y": "y",
      "yM": "M/y",
      "yMd": "d/M/y",
      "yMEd": "E, d/M/y",
      "yMM": "MM/y",
      "yMMM": "MMM y",
      "yMMMd": "d MMM, y",
      "yMMMED": "E, d MMM, y",
      "yMMMM": "MMMM y",
      "yQQQ": "QQQ y",

```

```

        "YQQQQ": "QQQQ Y",
        "yw": "الأسبوع w من سنة Y",
        "YW": "الأسبوع W من سنة Y",
        "yw": "الأسبوع w من سنة Y",
        "YW": "الأسبوع W من سنة Y",
        "yw": "الأسبوع w من سنة Y",
        "YW": "الأسبوع W من سنة Y",
        "yw": "الأسبوع w من سنة Y",
    },
    "appendItems": {
        "Day": "{0} ({2}: {1})",
        "Day-Of-Week": "{0} {1}",
        "Era": "{1} {0}",
        "Hour": "{0} ({2}: {1})",
        "Minute": "{0} ({2}: {1})",
        "Month": "{0} ({2}: {1})",
        "Quarter": "{0} ({2}: {1})",
        "Second": "{0} ({2}: {1})",
        "Timezone": "{0} {1}",
        "Week": "{0} ({2}: {1})",
        "Year": "{1} {0}"
    },
    "intervalFormats": {
        "intervalFormatFallback": "{0} - {1}",
        "d": {
            "d": "d-d"
        },
        "h": {
            "a": "h a - h a",
            "h": "h-h a"
        },
        "H": {
            "H": "HH-HH"
        },
        "hm": {
            "a": "h:mm a - h:mm a",
            "h": "h:mm-h:mm a",
            "m": "h:mm-h:mm a"
        },
        "Hm": {
            "H": "HH:mm-HH:mm",
            "m": "HH:mm-HH:mm"
        },
        "hmv": {
            "a": "h:mm a - h:mm a v",
            "h": "h:mm-h:mm a v",
            "m": "h:mm-h:mm a v"
        },
        "Hmv": {
            "H": "HH:mm-HH:mm v",
            "m": "HH:mm-HH:mm v"
        },
        "hv": {
            "a": "h a - h a v",
            "h": "h-h a v"
        },
        "Hv": {
            "H": "HH-HH v"
        }
    }

```

```

    },
    "M": {
        "M": "M-M"
    },
    "Md": {
        "d": "M/d - M/d",
        "M": "M/d - M/d"
    },
    "MEd": {
        "d": "E, d/M - E, d/M",
        "M": "E, d/M - E, d/M"
    },
    "MMM": {
        "M": "MMM-MMM"
    },
    "MMMd": {
        "d": "d-d MMM",
        "M": "d MMM - d MMM"
    },
    "MMMED": {
        "d": "E, d - E, d MMM",
        "M": "E, d MMM - E, d MMM"
    },
    "MMMM": {
        "M": "LLLL-LLLL"
    },
    "Y": {
        "Y": "Y-Y"
    },
    "YM": {
        "M": "M/Y - M/Y",
        "Y": "M/Y - M/Y"
    },
    "YMd": {
        "d": "d/M/Y - d/M/Y",
        "M": "d/M/Y - d/M/Y",
        "Y": "d/M/Y - d/M/Y"
    },
    "YMEd": {
        "d": "E, dd/MM/Y - E, dd/MM/Y",
        "M": "E, d/M/Y - E, d/M/Y",
        "Y": "E, d/M/Y - E, d/M/Y"
    },
    "YMMM": {
        "M": "MMM - MMM, Y",
        "Y": "MMM, Y - MMM, Y"
    },
    "YMMMd": {
        "d": "d-d MMM, Y",
        "M": "d MMM - d MMM, Y",
        "Y": "d MMM, Y - d MMM, Y"
    },
    "YMMMED": {
        "d": "E, d - E, d MMM, Y",
        "M": "E, d MMM - E, d MMM, Y",
        "Y": "E, d MMM, Y - E, d MMM, Y"
    },
    },

```



```

        "7";
        "يوليو",
        "8";
        "أغسطس",
        "9";
        "سبتمبر",
        "10";
        "أكتوبر",
        "11";
        "نوفمبر",
        "12";
        "ديسمبر";
    }
    "narrow";
    {
        "1";
        "ي",
        "2";
        "ف",
        "3";
        "م",
        "4";
        "أ",
        "5";
        "و",
        "6";
        "ن",
        "7";
        "ل",
        "8";
        "غ",
        "9";
        "س",
        "10";
        "ك",
        "11";
        "ب",
        "12";
        "د";
    }
    "wide";
    {
        "1";
        "يناير",
        "2";
        "فبراير",
        "3";
        "مارس",
        "4";
        "أبريل",
        "5";
        "مايو",
        "6";
        "يونيو",
        "7";
        "يوليو",
        "8";
    }

```

```

        "أغسطس",
        "9";
        "سبتمبر",
        "10";
        "أكتوبر",
        "11";
        "نوفمبر",
        "12";
        "ديسمبر";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
        "يناير",
        "2";
        "فبراير",
        "3";
        "مارس",
        "4";
        "أبريل",
        "5";
        "مايو",
        "6";
        "يونيو",
        "7";
        "يوليو",
        "8";
        "أغسطس",
        "9";
        "سبتمبر",
        "10";
        "أكتوبر",
        "11";
        "نوفمبر",
        "12";
        "ديسمبر";
    }
}
"narrow";
{
    "1";
    "ي",
    "2";
    "ف",
    "3";
    "م",
    "4";
    "أ",
    "5";
    "و",
    "6";
    "ن",
    "7";
    "ل",
    "8";
}

```

```

        "غ",
        "9";
        "س",
        "10";
        "ك",
        "11";
        "ب",
        "12";
        "د";
    }
    "wide";
    {
        "1";
        "يناير",
        "2";
        "فبراير",
        "3";
        "مارس",
        "4";
        "أبريل",
        "5";
        "مايو",
        "6";
        "يونيو",
        "7";
        "يوليو",
        "8";
        "أغسطس",
        "9";
        "سبتمبر",
        "10";
        "أكتوبر",
        "11";
        "نوفمبر",
        "12";
        "ديسمبر";
    }
}
"days";
{
    "format";
    {
        "abbreviated";
        {
            "sun";
            "الأحد",
            "mon";
            "الاثنين",
            "tue";
            "الثلاثاء",
            "wed";
            "الأربعاء",
            "thu";
            "الخميس",
            "fri";
            "الجمعة",

```

```

        "sat";
        "السبت";
    }
    "narrow";
    {
        "sun";
        "ح", "mon";
        "ن", "tue";
        "ث", "wed";
        "ر", "thu";
        "خ", "fri";
        "ج", "sat";
        "س";
    }
    "short";
    {
        "sun";
        "الأحد", "mon";
        "الاثنين", "tue";
        "الثلاثاء", "wed";
        "الأربعاء", "thu";
        "الخميس", "fri";
        "الجمعة", "sat";
        "السبت";
    }
    "wide";
    {
        "sun";
        "الأحد", "mon";
        "الاثنين", "tue";
        "الثلاثاء", "wed";
        "الأربعاء", "thu";
        "الخميس", "fri";
        "الجمعة", "sat";
        "السبت";
    }
    }
    "stand-alone";
    {

```



```

"abbreviated";
{
  "sun";
  "الأحد",
  "mon";
  "الاثنين",
  "tue";
  "الثلاثاء",
  "wed";
  "الأربعاء",
  "thu";
  "الخميس",
  "fri";
  "الجمعة",
  "sat";
  "السبت";
}
"narrow";
{
  "sun";
  "ح",
  "mon";
  "ن",
  "tue";
  "ث",
  "wed";
  "ر",
  "thu";
  "خ",
  "fri";
  "ج",
  "sat";
  "س";
}
"short";
{
  "sun";
  "الأحد",
  "mon";
  "الاثنين",
  "tue";
  "الثلاثاء",
  "wed";
  "الأربعاء",
  "thu";
  "الخميس",
  "fri";
  "الجمعة",
  "sat";
  "السبت";
}
"wide";
{
  "sun";
  "الأحد",
  "mon";
  "الاثنين",

```

```

        "tue";
        "الثلاثاء",
        "wed";
        "الأربعاء",
        "thu";
        "الخميس",
        "fri";
        "الجمعة",
        "sat";
        "السبت";
    }
}
}
"quarters";
{
    "format";
    {
        "abbreviated";
        {
            "1";
            "الربع الأول",
            "2";
            "الربع الثاني",
            "3";
            "الربع الثالث",
            "4";
            "الربع الرابع";
        }
        "narrow";
        {
            "1";
            "١",
            "2";
            "٢",
            "3";
            "٣",
            "4";
            "٤";
        }
        "wide";
        {
            "1";
            "الربع الأول",
            "2";
            "الربع الثاني",
            "3";
            "الربع الثالث",
            "4";
            "الربع الرابع";
        }
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
        "الربع الأول",

```

```

        "2";
        "الربع الثاني",
        "3";
        "الربع الثالث",
        "4";
        "الربع الرابع";
    }
    "narrow";
    {
        "1";
        "١",
        "2";
        "٢",
        "3";
        "٣",
        "4";
        "٤";
    }
    "wide";
    {
        "1";
        "الربع الأول",
        "2";
        "الربع الثاني",
        "3";
        "الربع الثالث",
        "4";
        "الربع الرابع";
    }
    }
}
"dayPeriods";
{
    "format";
    {
        "abbreviated";
        {
            "am";
            "ص",
            "pm";
            "م",
            "morning1";
            "فجرا",
            "morning2";
            "ص",
            "afternoon1";
            "ظهراً",
            "afternoon2";
            "بعد الظهر",
            "evening1";
            "مساء",
            "night1";
            "منتصف الليل",
            "night2";
            "ل";
        }
    }
    "narrow";

```

```

    {
      "am";
      "ص",
      "pm";
      "م",
      "morning1";
      "فجرًا",
      "morning2";
      "صباحًا",
      "afternoon1";
      "ظهرًا",
      "afternoon2";
      "بعد الظهر",
      "evening1";
      "مساءً",
      "night1";
      "منتصف الليل",
      "night2";
      "ليلاً";
    }
    "wide";
    {
      "am";
      "ص",
      "pm";
      "م",
      "morning1";
      "فجرًا",
      "morning2";
      "صباحًا",
      "afternoon1";
      "ظهرًا",
      "afternoon2";
      "بعد الظهر",
      "evening1";
      "مساءً",
      "night1";
      "منتصف الليل",
      "night2";
      "ليلاً";
    }
  }
  "stand-alone";
  {
    "abbreviated";
    {
      "am";
      "ص",
      "pm";
      "م",
      "morning1";
      "فجرًا",
      "morning2";
      "ص",
      "afternoon1";
      "ظهرًا",
      "afternoon2";
    }
  }

```

```

        "بعد الظهر",
        "evening1";
        "مساءً",
        "night1";
        "منتصف الليل",
        "night2";
        "ليلاً";
    }
    "narrow";
    {
        "am";
        "ص",
        "pm";
        "م",
        "morning1";
        "فجراً",
        "morning2";
        "صباحاً",
        "afternoon1";
        "ظهراً",
        "afternoon2";
        "بعد الظهر",
        "evening1";
        "مساءً",
        "night1";
        "منتصف الليل",
        "night2";
        "ليلاً";
    }
    "wide";
    {
        "am";
        "صباحاً",
        "pm";
        "مساءً",
        "morning1";
        "فجراً",
        "morning2";
        "صباحاً",
        "afternoon1";
        "ظهراً",
        "afternoon2";
        "بعد الظهر",
        "evening1";
        "مساءً",
        "night1";
        "منتصف الليل",
        "night2";
        "ليلاً";
    }
    }
    "eras";
    {
        "eraNames";
        {
            "0";

```

```

        "قبل الميلاد",
        "0-alt-variant";
        "BCE",
        "1";
        "ميلادي",
        "1-alt-variant";
        "بعد الميلاد";
    }
    "eraAbbr";
    {
        "0";
        "ق.م",
        "0-alt-variant";
        "BCE",
        "1";
        "م",
        "1-alt-variant";
        "م.ب";
    }
    "eraNarrow";
    {
        "0";
        "ق.م",
        "0-alt-variant";
        "BCE",
        "1";
        "م",
        "1-alt-variant";
        "م.ب";
    }
}
"dateFormats";
{
    "full";
    "EEEE, d MMMM, y",
    "long";
    "d MMMM, y",
    "medium";
    "dd/MM/y",
    "short";
    "d/M/y";
}
"timeFormats";
{
    "full";
    "h:mm:ss a zzzz",
    "long";
    "h:mm:ss a z",
    "medium";
    "h:mm:ss a",
    "short";
    "h:mm a";
}
"dateTimeFormats";
{
    "full";
    "{1} {0}",

```

```

        "long";
    "{1} {0}",
        "medium";
    "{1} {0}",
        "short";
    "{1} {0}",
        "availableFormats";
    {
        "d";
        "d",
            "E";
        "ccc",
            "Ed";
        "E, d",
            "Ehm";
        "E h:mm a",
            "EHm";
        "E HH:mm",
            "Ehms";
        "E h:mm:ss a",
            "EHms";
        "E HH:mm:ss",
            "Gy";
        "y G",
            "GyMMM";
        "MMM y G",
            "GyMMMd";
        "d MMM, y G",
            "GyMMMED";
        "E, d MMM, y G",
            "h";
        "h a",
            "H";
        "HH",
            "hm";
        "h:mm a",
            "Hm";
        "HH:mm",
            "hms";
        "h:mm:ss a",
            "Hms";
        "HH:mm:ss",
            "hmsv";
        "h:mm:ss a v",
            "Hmsv";
        "HH:mm:ss v",
            "hmv";
        "h:mm a v",
            "Hmv";
        "HH:mm v",
            "M";
        "L",
            "Md";
        "d/M",
            "MEd";
        "E, d/M",
            "MMdd";
    }

```

```

"dd/MM",
  "MMM";
"LLL",
  "MMMd";
"d MMM",
  "MMMEd";
"E, d MMM",
  "MMMMd";
"d MMMM",
  "MMMMEd";
"E, d MMMM",
  "MMMMW";
"من الاسبوع W MMM",
  "MMMMW";
"من الاسبوع W MMM",
  "MMMMW";
"من الاسبوع W MMM",
  "MMMMW";
"من الاسبوع W MMM",
  "MMMMW";
"من الاسبوع W MMM",
  "MMMMW";
"من الاسبوع W MMM",
  "MMMMW";
"من الاسبوع W MMM",
  "ms";
"mm:ss",
  "y";
"y",
  "yM";
"M/y",
  "yMd";
"d/M/y",
  "yMEd";
"E, d/M/y",
  "yMM";
"MM/y",
  "yMMM";
"MMM y",
  "yMMMd";
"d MMM, y",
  "yMMMEd";
"E, d MMM, y",
  "yMMMM";
"MMMM y",
  "yQQQ";
"QQQ y",
  "yQQQQ";
"QQQQ y",
  "yw";
"y من سنة w الاسبوع",
  "yw";
"y من سنة w الاسبوع",
  "yw";
"y من سنة w الاسبوع",
  "yw";
"y من سنة w الاسبوع",
  "yw";
"y من سنة w الاسبوع",
  "yw";
"y من سنة w الاسبوع",
  "yw";

```



```

        "yw";
        "y من سنة w الأسبوع";
    }
    "appendItems";
    {
        "Day";
        "{0} ({2}: {1})",
        "Day-Of-Week";
        "{0} {1}",
        "Era";
        "{1} {0}",
        "Hour";
        "{0} ({2}: {1})",
        "Minute";
        "{0} ({2}: {1})",
        "Month";
        "{0} ({2}: {1})",
        "Quarter";
        "{0} ({2}: {1})",
        "Second";
        "{0} ({2}: {1})",
        "Timezone";
        "{0} {1}",
        "Week";
        "{0} ({2}: {1})",
        "Year";
        "{1} {0}";
    }
    "intervalFormats";
    {
        "intervalFormatFallback";
        "{0} - {1}",
        "d";
        {
            "d";
            "d-d";
        }
        "h";
        {
            "a";
            "h a - h a",
            "h";
            "h-h a";
        }
        "H";
        {
            "H";
            "HH-HH";
        }
        "hm";
        {
            "a";
            "h:mm a - h:mm a",
            "h";
            "h:mm-h:mm a",
            "m";
            "h:mm-h:mm a";
        }
    }

```

```

    }
    "Hm";
    {
        "H";
        "HH:mm-HH:mm",
        "m";
        "HH:mm-HH:mm";
    }
    "hmv";
    {
        "a";
        "h:mm a - h:mm a v",
        "h";
        "h:mm-h:mm a v",
        "m";
        "h:mm-h:mm a v";
    }
    "Hmv";
    {
        "H";
        "HH:mm-HH:mm v",
        "m";
        "HH:mm-HH:mm v";
    }
    "hv";
    {
        "a";
        "h a - h a v",
        "h";
        "h-h a v";
    }
    "Hv";
    {
        "H";
        "HH-HH v";
    }
    "M";
    {
        "M";
        "M-M";
    }
    "Md";
    {
        "d";
        "M/d - M/d",
        "M";
        "M/d - M/d";
    }
    "MEd";
    {
        "d";
        "E, d/M - E, d/M",
        "M";
        "E, d/M - E, d/M";
    }
    "MMM";
    {

```

```

        "M";
        "MMM-MMM";
    }
    "MMMd";
    {
        "d";
        "d-d MMM",
        "M";
        "d MMM - d MMM";
    }
    "MMMED";
    {
        "d";
        "E, d - E, d MMM",
        "M";
        "E, d MMM - E, d MMM";
    }
    "MMMM";
    {
        "M";
        "LLLL-LLLL";
    }
    "Y";
    {
        "Y";
        "Y-Y";
    }
    "YM";
    {
        "M";
        "M/Y - M/Y",
        "Y";
        "M/Y - M/Y";
    }
    "YMd";
    {
        "d";
        "d/M/Y - d/M/Y",
        "M";
        "d/M/Y - d/M/Y",
        "Y";
        "d/M/Y - d/M/Y";
    }
    "yMED";
    {
        "d";
        "E, dd/MM/Y - E, dd/MM/Y",
        "M";
        "E, d/M/Y - E, d/M/Y",
        "Y";
        "E, d/M/Y - E, d/M/Y";
    }
    "yMMM";
    {
        "M";
        "MMM - MMM, Y",
        "Y";
    }

```

```

}
    "MMM. y - MMM. y";
}
    "yMMMd";
{
        "d";
        "d-d MMM. y",
            "M";
        "d MMM - d MMM. y",
            "Y";
        "d MMM. y - d MMM. y";
    }
    "yMMMMd";
{
        "d";
        "E. d - E. d MMM. y",
            "M";
        "E. d MMM - E. d MMM. y",
            "Y";
        "E. d MMM. y - E. d MMM. y";
    }
    "yMMMM";
{
        "M";
        "MMMM - MMMM. y",
            "Y";
        "MMMM. y - MMMM. y";
    }
}
}
}
}
}
}
}
}
}
}

```

CURRENCIES.JSON

```
{
  "main": {
    "ar": {
      "identity": {
        "version": {
          "_number": "$Revision: 12879 $",
          "_cldrVersion": "30.0.3"
        },
        "language": "ar"
      },
      "numbers": {
        "currencies": {
          "ADP": {
            "displayName": "بيستا أندوري",
            "symbol": "ADP"
          },
          "AED": {
            "displayName": "درهم إماراتي",

```

```

    "displayName-count-zero": "درهم إماراتي",
    "displayName-count-one": "درهم إماراتي",
    "displayName-count-two": "درهم إماراتي",
    "displayName-count-few": "درهم إماراتي",
    "displayName-count-many": "درهم إماراتي",
    "displayName-count-other": "درهم إماراتي",
    "symbol": "د.إ."
  },
  "AFA": {
    "displayName": "2002-1927 - أفغاني",
    "symbol": "AFA"
  },
  "AFN": {
    "displayName": "أفغاني",
    "displayName-count-zero": "أفغاني أفغانستان",
    "displayName-count-one": "أفغاني أفغانستان",
    "displayName-count-two": "أفغاني أفغانستان",
    "displayName-count-few": "أفغاني أفغانستان",
    "displayName-count-many": "أفغاني أفغانستان",
    "displayName-count-other": "أفغاني أفغانستان",
    "symbol": "AFN"
  },
  "ALL": {
    "displayName": "ليك ألباني",
    "displayName-count-zero": "ليك ألباني",
    "displayName-count-one": "ليك ألباني",
    "displayName-count-two": "ليك ألباني",
    "displayName-count-few": "ليك ألباني",
    "displayName-count-many": "ليك ألباني",
    "displayName-count-other": "ليك ألباني",
    "symbol": "ALL"
  },
  "AMD": {
    "displayName": "درام أرميني",
    "displayName-count-zero": "درام أرميني",
    "displayName-count-one": "درام أرميني",
    "displayName-count-two": "درام أرميني",
    "displayName-count-few": "درام أرميني",
    "displayName-count-many": "درام أرميني",
    "displayName-count-other": "درام أرميني",
    "symbol": "AMD"
  },
  "ANG": {
    "displayName": "غيلدر أنتيلي هولندي",
    "displayName-count-zero": "غيلدر أنتيلي هولندي",
    "displayName-count-one": "غيلدر أنتيلي هولندي",
    "displayName-count-two": "غيلدر أنتيلي هولندي",
    "displayName-count-few": "غيلدر أنتيلي هولندي",
    "displayName-count-many": "غيلدر أنتيلي هولندي",
    "displayName-count-other": "غيلدر أنتيلي هولندي",
    "symbol": "ANG"
  },
  "AOA": {
    "displayName": "كوانزا أنجولي",
    "displayName-count-zero": "كوانزا أنجولي",
    "displayName-count-one": "كوانزا أنجولي",
    "displayName-count-two": "كوانزا أنجولي",

```

```

        "displayName-count-few": "كوانزا أنجولي",
        "displayName-count-many": "كوانزا أنجولي",
        "displayName-count-other": "كوانزا أنجولي",
        "symbol": "AOA",
        "symbol-alt-narrow": "Kz"
    },
    "AOK": {
        "displayName": "1990-1977 - كوانزا أنجولي",
        "symbol": "AOK"
    },
    "AON": {
        "displayName": "2000-1990 - كوانزا أنجولي جديدة",
        "symbol": "AON"
    },
    "AOR": {
        "displayName": "1999 - 1995 - كوانزا أنجولي معدلة",
        "symbol": "AOR"
    },
    "ARA": {
        "displayName": "استرال أرجنتيني",
        "symbol": "ARA"
    },
    "ARL": {
        "displayName": "ARL",
        "symbol": "ARL"
    },
    "ARM": {
        "displayName": "ARM",
        "symbol": "ARM"
    },
    "ARP": {
        "displayName": "1985-1983 - بيزو أرجنتيني",
        "symbol": "ARP"
    },
    "ARS": {
        "displayName": "بيزو أرجنتيني",
        "displayName-count-zero": "بيزو أرجنتيني",
        "displayName-count-one": "بيزو أرجنتيني",
        "displayName-count-two": "بيزو أرجنتيني",
        "displayName-count-few": "بيزو أرجنتيني",
        "displayName-count-many": "بيزو أرجنتيني",
        "displayName-count-other": "بيزو أرجنتيني",
        "symbol": "ARS",
        "symbol-alt-narrow": "AR$"
    },
    "ATS": {
        "displayName": "شلن نمساوي",
        "symbol": "ATS"
    },
    "AUD": {
        "displayName": "دولار أسترالي",
        "displayName-count-zero": "دولار أسترالي",
        "displayName-count-one": "دولار أسترالي",
        "displayName-count-two": "دولار أسترالي",
        "displayName-count-few": "دولار أسترالي",
        "displayName-count-many": "دولار أسترالي",
        "displayName-count-other": "دولار أسترالي",

```

```

    "symbol": "AU$",
    "symbol-alt-narrow": "AU$"
  },
  "AWG": {
    "displayName": "فلورن أروبي",
    "displayName-count-zero": "فلورن أروبي",
    "displayName-count-one": "فلورن أروبي",
    "displayName-count-two": "فلورن أروبي",
    "displayName-count-few": "فلورن أروبي",
    "displayName-count-many": "فلورن أروبي",
    "displayName-count-other": "فلورن أروبي",
    "symbol": "AWG"
  },
  "AZM": {
    "displayName": "مانات أذربيجاني",
    "symbol": "AZM"
  },
  "AZN": {
    "displayName": "مانات أذربيجان",
    "displayName-count-zero": "مانت أذربيجاني",
    "displayName-count-one": "مانت أذربيجاني",
    "displayName-count-two": "مانت أذربيجاني",
    "displayName-count-few": "مانت أذربيجاني",
    "displayName-count-many": "مانت أذربيجاني",
    "displayName-count-other": "مانت أذربيجاني",
    "symbol": "AZN"
  },
  "BAD": {
    "displayName": "دينار البوسنة والهرسك",
    "symbol": "BAD"
  },
  "BAM": {
    "displayName": "مارك البوسنة والهرسك قابل للتحويل",
    "displayName-count-zero": "مارك البوسنة والهرسك قابل للتحويل",
    "displayName-count-one": "مارك البوسنة والهرسك قابل للتحويل",
    "displayName-count-two": "مارك البوسنة والهرسك قابل للتحويل",
    "displayName-count-few": "مارك البوسنة والهرسك قابل للتحويل",
    "displayName-count-many": "مارك البوسنة والهرسك قابل للتحويل",
    "displayName-count-other": "مارك البوسنة والهرسك قابل للتحويل",
    "symbol": "BAM",
    "symbol-alt-narrow": "KM"
  },
  "BAN": {
    "displayName": "BAN",
    "symbol": "BAN"
  },
  "BBD": {
    "displayName": "دولار بربادوسي",
    "displayName-count-zero": "دولار بربادوسي",
    "displayName-count-one": "دولار بربادوسي",
    "displayName-count-two": "دولار بربادوسي",
    "displayName-count-few": "دولار بربادوسي",
    "displayName-count-many": "دولار بربادوسي",
    "displayName-count-other": "دولار بربادوسي",
    "symbol": "BBD",
    "symbol-alt-narrow": "BB$"
  },
}

```

```

"BDT": {
  "displayName": "তাকা بنجلاديشي",
  "displayName-count-zero": "تاکا بنجلاديشي",
  "displayName-count-one": "تاکا بنجلاديشي",
  "displayName-count-two": "تاکا بنجلاديشي",
  "displayName-count-few": "تاکا بنجلاديشي",
  "displayName-count-many": "تاکا بنجلاديشي",
  "displayName-count-other": "تاکا بنجلاديشي",
  "symbol": "BDT",
  "symbol-alt-narrow": "₳"
},
"BEC": {
  "displayName": "فرنك بلجيكي قابل للتحويل",
  "symbol": "BEC"
},
"BEF": {
  "displayName": "فرنك بلجيكي",
  "symbol": "BEF"
},
"BEL": {
  "displayName": "فرنك بلجيكي مالي",
  "symbol": "BEL"
},
"BGL": {
  "displayName": "BGL",
  "symbol": "BGL"
},
"BGM": {
  "displayName": "BGM",
  "symbol": "BGM"
},
"BGN": {
  "displayName": "ليف بلغاري",
  "displayName-count-zero": "ليف بلغاري",
  "displayName-count-one": "ليف بلغاري",
  "displayName-count-two": "ليف بلغاري",
  "displayName-count-few": "ليف بلغاري",
  "displayName-count-many": "ليف بلغاري",
  "displayName-count-other": "ليف بلغاري",
  "symbol": "BGN"
},
"BGO": {
  "displayName": "BGO",
  "symbol": "BGO"
},
"BHD": {
  "displayName": "دينار بحريني",
  "displayName-count-zero": "دينار بحريني",
  "displayName-count-one": "دينار بحريني",
  "displayName-count-two": "دينار بحريني",
  "displayName-count-few": "دينار بحريني",
  "displayName-count-many": "دينار بحريني",
  "displayName-count-other": "دينار بحريني",
  "symbol": "د.ب."
},
"BIF": {

```



```

    "displayName": "فرنك بروندي",
    "displayName-count-zero": "فرنك بروندي",
    "displayName-count-one": "فرنك بروندي",
    "displayName-count-two": "فرنك بروندي",
    "displayName-count-few": "فرنك بروندي",
    "displayName-count-many": "فرنك بروندي",
    "displayName-count-other": "فرنك بروندي",
    "symbol": "BIF"
  },
  "BMD": {
    "displayName": "دولار برمودي",
    "displayName-count-zero": "دولار برمودي",
    "displayName-count-one": "دولار برمودي",
    "displayName-count-two": "دولار برمودي",
    "displayName-count-few": "دولار برمودي",
    "displayName-count-many": "دولار برمودي",
    "displayName-count-other": "دولار برمودي",
    "symbol": "BMD",
    "symbol-alt-narrow": "BM$"
  },
  "BND": {
    "displayName": "دولار بروناي",
    "displayName-count-zero": "دولار بروناي",
    "displayName-count-one": "دولار بروناي",
    "displayName-count-two": "دولار بروناي",
    "displayName-count-few": "دولار بروناي",
    "displayName-count-many": "دولار بروناي",
    "displayName-count-other": "دولار بروناي",
    "symbol": "BND",
    "symbol-alt-narrow": "BN$"
  },
  "BOB": {
    "displayName": "بوليفيانو بوليفي",
    "displayName-count-zero": "بوليفيانو بوليفي",
    "displayName-count-one": "بوليفيانو بوليفي",
    "displayName-count-two": "بوليفيانو بوليفي",
    "displayName-count-few": "بوليفيانو بوليفي",
    "displayName-count-many": "بوليفيانو بوليفي",
    "displayName-count-other": "بوليفيانو بوليفي",
    "symbol": "BOB",
    "symbol-alt-narrow": "Bs"
  },
  "BOL": {
    "displayName": "BOL",
    "symbol": "BOL"
  },
  "BOP": {
    "displayName": "بيزو بوليفي",
    "symbol": "BOP"
  },
  "BOV": {
    "displayName": "مفدول بوليفي",
    "symbol": "BOV"
  },
  "BRB": {
    "displayName": "نوفو كروزايرو برازيلي - 1986-1967",
    "symbol": "BRB"
  }

```

```

    },
    "BRC": {
      "displayName": "کروزادو برازیلی",
      "symbol": "BRC"
    },
    },
    "BRE": {
      "displayName": "کروزایرو برازیلی - 1993-1990",
      "symbol": "BRE"
    },
    },
    "BRL": {
      "displayName": "ریال برازیلی",
      "displayName-count-zero": "ریال برازیلی",
      "displayName-count-one": "ریال برازیلی",
      "displayName-count-two": "ریال برازیلی",
      "displayName-count-few": "ریال برازیلی",
      "displayName-count-many": "ریال برازیلی",
      "displayName-count-other": "ریال برازیلی",
      "symbol": "R$",
      "symbol-alt-narrow": "R$"
    },
    },
    "BRN": {
      "displayName": "BRN",
      "symbol": "BRN"
    },
    },
    "BRR": {
      "displayName": "BRR",
      "symbol": "BRR"
    },
    },
    "BRZ": {
      "displayName": "BRZ",
      "symbol": "BRZ"
    },
    },
    "BSD": {
      "displayName": "دولار باهامی",
      "displayName-count-zero": "دولار باهامی",
      "displayName-count-one": "دولار باهامی",
      "displayName-count-two": "دولار باهامی",
      "displayName-count-few": "دولار باهامی",
      "displayName-count-many": "دولار باهامی",
      "displayName-count-other": "دولار باهامی",
      "symbol": "BSD",
      "symbol-alt-narrow": "BS$"
    },
    },
    "BTN": {
      "displayName": "نولتوم بوتانی",
      "displayName-count-zero": "نولتوم بوتانی",
      "displayName-count-one": "نولتوم بوتانی",
      "displayName-count-two": "نولتوم بوتانی",
      "displayName-count-few": "نولتوم بوتانی",
      "displayName-count-many": "نولتوم بوتانی",
      "displayName-count-other": "نولتوم بوتانی",
      "symbol": "BTN"
    },
    },
    "BUK": {
      "displayName": "کیات بورمی",
      "symbol": "BUK"
    },
    },
  },

```

```

"BWP": {
  "displayName": "بولا بتسواني",
  "displayName-count-zero": "بولا بتسواني",
  "displayName-count-one": "بولا بتسواني",
  "displayName-count-two": "بولا بتسواني",
  "displayName-count-few": "بولا بتسواني",
  "displayName-count-many": "بولا بتسواني",
  "displayName-count-other": "بولا بتسواني",
  "symbol": "BWP",
  "symbol-alt-narrow": "P"
},
"BYB": {
  "displayName": "روبل بيلاروسي جديد - 1999-1994",
  "symbol": "BYB"
},
"BYN": {
  "displayName": "روبل بيلاروسي",
  "displayName-count-zero": "روبل بيلاروسي",
  "displayName-count-one": "روبل بيلاروسي",
  "displayName-count-two": "روبل بيلاروسي",
  "displayName-count-few": "روبل بيلاروسي",
  "displayName-count-many": "روبل بيلاروسي",
  "displayName-count-other": "روبل بيلاروسي",
  "symbol": "BYN",
  "symbol-alt-narrow": "p."
},
"BYR": {
  "displayName": "روبل بيلاروسي (2016-2000)",
  "displayName-count-zero": "روبل بيلاروسي (2016-2000)",
  "displayName-count-one": "روبل بيلاروسي (2016-2000)",
  "displayName-count-two": "روبل بيلاروسي (2016-2000)",
  "displayName-count-few": "روبل بيلاروسي (2016-2000)",
  "displayName-count-many": "روبل بيلاروسي (2016-2000)",
  "displayName-count-other": "روبل بيلاروسي (2016-2000)",
  "symbol": "BYR"
},
"BZD": {
  "displayName": "دولار بليزي",
  "displayName-count-zero": "دولار بليزي",
  "displayName-count-one": "دولار بليزي",
  "displayName-count-two": "دولاران بليزيان",
  "displayName-count-few": "دولار بليزي",
  "displayName-count-many": "دولار بليزي",
  "displayName-count-other": "دولار بليزي",
  "symbol": "BZD",
  "symbol-alt-narrow": "BZ$"
},
"CAD": {
  "displayName": "دولار كندي",
  "displayName-count-zero": "دولار كندي",
  "displayName-count-one": "دولار كندي",
  "displayName-count-two": "دولار كندي",
  "displayName-count-few": "دولار كندي",
  "displayName-count-many": "دولار كندي",
  "displayName-count-other": "دولار كندي",
  "symbol": "CA$",
  "symbol-alt-narrow": "CA$"
}

```

```

    },
    "CDF": {
      "displayName": "فرنك كونغولي",
      "displayName-count-zero": "فرنك كونغولي",
      "displayName-count-one": "فرنك كونغولي",
      "displayName-count-two": "فرنك كونغولي",
      "displayName-count-few": "فرنك كونغولي",
      "displayName-count-many": "فرنك كونغولي",
      "displayName-count-other": "فرنك كونغولي",
      "symbol": "CDF"
    },
    "CHE": {
      "displayName": "CHE",
      "symbol": "CHE"
    },
    "CHF": {
      "displayName": "فرنك سويسري",
      "displayName-count-zero": "فرنك سويسري",
      "displayName-count-one": "فرنك سويسري",
      "displayName-count-two": "فرنك سويسري",
      "displayName-count-few": "فرنك سويسري",
      "displayName-count-many": "فرنك سويسري",
      "displayName-count-other": "فرنك سويسري",
      "symbol": "CHF"
    },
    "CHW": {
      "displayName": "CHW",
      "symbol": "CHW"
    },
    "CLE": {
      "displayName": "CLE",
      "symbol": "CLE"
    },
    "CLF": {
      "displayName": "CLF",
      "symbol": "CLF"
    },
    "CLP": {
      "displayName": "بيزو شيلي",
      "displayName-count-zero": "بيزو شيلي",
      "displayName-count-one": "بيزو شيلي",
      "displayName-count-two": "بيزو شيلي",
      "displayName-count-few": "بيزو شيلي",
      "displayName-count-many": "بيزو شيلي",
      "displayName-count-other": "بيزو شيلي",
      "symbol": "CLP",
      "symbol-alt-narrow": "CL$"
    },
    "CNY": {
      "displayName": "يوان صيني",
      "displayName-count-zero": "يوان صيني",
      "displayName-count-one": "يوان صيني",
      "displayName-count-two": "يوان صيني",
      "displayName-count-few": "يوان صيني",
      "displayName-count-many": "يوان صيني",
      "displayName-count-other": "يوان صيني",
      "symbol": "CN¥",

```

```

    "symbol-alt-narrow": "CN¥"
  },
  "COP": {
    "displayName": "بيزو كولومبي",
    "displayName-count-zero": "بيزو كولومبي",
    "displayName-count-one": "بيزو كولومبي",
    "displayName-count-two": "بيزو كولومبي",
    "displayName-count-few": "بيزو كولومبي",
    "displayName-count-many": "بيزو كولومبي",
    "displayName-count-other": "بيزو كولومبي",
    "symbol": "COP",
    "symbol-alt-narrow": "CO$"
  },
  "COU": {
    "displayName": "COU",
    "symbol": "COU"
  },
  "CRC": {
    "displayName": "كولن كوستا ريكي",
    "displayName-count-zero": "كولن كوستا ريكي",
    "displayName-count-one": "كولن كوستا ريكي",
    "displayName-count-two": "كولن كوستا ريكي",
    "displayName-count-few": "كولن كوستا ريكي",
    "displayName-count-many": "كولن كوستا ريكي",
    "displayName-count-other": "كولن كوستا ريكي",
    "symbol": "CRC",
    "symbol-alt-narrow": "₡"
  },
  "CSD": {
    "displayName": "دينار صربي قديم",
    "symbol": "CSD"
  },
  "CSK": {
    "displayName": "كروننة تشيكوسلوفاكيا",
    "symbol": "CSK"
  },
  "CUC": {
    "displayName": "بيزو كوبي قابل للتحويل",
    "displayName-count-zero": "بيزو كوبي قابل للتحويل",
    "displayName-count-one": "بيزو كوبي قابل للتحويل",
    "displayName-count-two": "بيزو كوبي قابل للتحويل",
    "displayName-count-few": "بيزو كوبي قابل للتحويل",
    "displayName-count-many": "بيزو كوبي قابل للتحويل",
    "displayName-count-other": "بيزو كوبي قابل للتحويل",
    "symbol": "CUC",
    "symbol-alt-narrow": "$"
  },
  "CUP": {
    "displayName": "بيزو كوبي",
    "displayName-count-zero": "بيزو كوبي",
    "displayName-count-one": "بيزو كوبي",
    "displayName-count-two": "بيزو كوبي",
    "displayName-count-few": "بيزو كوبي",
    "displayName-count-many": "بيزو كوبي",
    "displayName-count-other": "بيزو كوبي",
    "symbol": "CUP",
    "symbol-alt-narrow": "CU$"
  }

```

```

    },
    "CVE": {
      "displayName": "اسكودو الرأس الخضراء",
      "displayName-count-zero": "اسكودو الرأس الخضراء",
      "displayName-count-one": "اسكودو الرأس الخضراء",
      "displayName-count-two": "اسكودو الرأس الخضراء",
      "displayName-count-few": "اسكودو الرأس الخضراء",
      "displayName-count-many": "اسكودو الرأس الخضراء",
      "displayName-count-other": "اسكودو الرأس الخضراء",
      "symbol": "CVE"
    },
    "CYP": {
      "displayName": "جنيه قبرصي",
      "symbol": "CYP"
    },
    "CZK": {
      "displayName": "كرونة تشيكية",
      "displayName-count-zero": "كرونة تشيكية",
      "displayName-count-one": "كرونة تشيكية",
      "displayName-count-two": "كرونة تشيكية",
      "displayName-count-few": "كرونة تشيكية",
      "displayName-count-many": "كرونة تشيكية",
      "displayName-count-other": "كرونة تشيكية",
      "symbol": "CZK",
      "symbol-alt-narrow": "Kč"
    },
    "DDM": {
      "displayName": "أوستمارك ألماني شرقي",
      "symbol": "DDM"
    },
    "DEM": {
      "displayName": "مارك ألماني",
      "symbol": "DEM"
    },
    "DJF": {
      "displayName": "فرنك جيبوتي",
      "displayName-count-zero": "فرنك جيبوتي",
      "displayName-count-one": "فرنك جيبوتي",
      "displayName-count-two": "فرنك جيبوتي",
      "displayName-count-few": "فرنك جيبوتي",
      "displayName-count-many": "فرنك جيبوتي",
      "displayName-count-other": "فرنك جيبوتي",
      "symbol": "DJF"
    },
    "DKK": {
      "displayName": "كرونة دانماركي",
      "displayName-count-zero": "كرونة دانماركي",
      "displayName-count-one": "كرونة دانماركي",
      "displayName-count-two": "كرونة دانماركي",
      "displayName-count-few": "كرونة دانماركي",
      "displayName-count-many": "كرونة دانماركي",
      "displayName-count-other": "كرونة دانماركي",
      "symbol": "DKK",
      "symbol-alt-narrow": "kr"
    },
    "DOP": {
      "displayName": "بيزو الدومنيكان",

```

```

    "displayName-count-zero": "بیزو الدومنیکان",
    "displayName-count-one": "بیزو الدومنیکان",
    "displayName-count-two": "بیزو الدومنیکان",
    "displayName-count-few": "بیزو الدومنیکان",
    "displayName-count-many": "بیزو الدومنیکان",
    "displayName-count-other": "بیزو الدومنیکان",
    "symbol": "DOP",
    "symbol-alt-narrow": "DO$"
  },
  "DZD": {
    "displayName": "دینار جزائري",
    "displayName-count-zero": "دینار جزائري",
    "displayName-count-one": "دینار جزائري",
    "displayName-count-two": "دیناران جزائريان",
    "displayName-count-few": "دینارات جزائرية",
    "displayName-count-many": "دینارًا جزائريًا",
    "displayName-count-other": "دینار جزائري",
    "symbol": "د.ج."
  },
  "ECS": {
    "displayName": "ECS",
    "symbol": "ECS"
  },
  "ECV": {
    "displayName": "ECV",
    "symbol": "ECV"
  },
  "EEK": {
    "displayName": "کرونة استونية",
    "symbol": "EEK"
  },
  "EGP": {
    "displayName": "جنيه مصري",
    "displayName-count-zero": "جنيه مصري",
    "displayName-count-one": "جنيه مصري",
    "displayName-count-two": "جنيهان مصريان",
    "displayName-count-few": "جنيهات مصرية",
    "displayName-count-many": "جنيهاً مصريًا",
    "displayName-count-other": "جنيه مصري",
    "symbol": "ج.م.",
    "symbol-alt-narrow": "E£"
  },
  "ERN": {
    "displayName": "ناكفا أريتري",
    "displayName-count-zero": "ناكفا أريتري",
    "displayName-count-one": "ناكفا أريتري",
    "displayName-count-two": "ناكفا أريتري",
    "displayName-count-few": "ناكفا أريتري",
    "displayName-count-many": "ناكفا أريتري",
    "displayName-count-other": "ناكفا أريتري",
    "symbol": "ERN"
  },
  "ESA": {
    "displayName": "ESA",
    "symbol": "ESA"
  },
  "ESB": {

```

```

    "displayName": "ESB",
    "symbol": "ESB"
  },
  "ESP": {
    "displayName": "بیزیتا إسبانی",
    "symbol": "ESP",
    "symbol-alt-narrow": "₧"
  },
  "ETB": {
    "displayName": "بیر أثیوبی",
    "displayName-count-zero": "بیر أثیوبی",
    "displayName-count-one": "بیر أثیوبی",
    "displayName-count-two": "بیر أثیوبی",
    "displayName-count-few": "بیر أثیوبی",
    "displayName-count-many": "بیر أثیوبی",
    "displayName-count-other": "بیر أثیوبی",
    "symbol": "ETB"
  },
  "EUR": {
    "displayName": "یورو",
    "displayName-count-zero": "یورو",
    "displayName-count-one": "یورو",
    "displayName-count-two": "یورو",
    "displayName-count-few": "یورو",
    "displayName-count-many": "یورو",
    "displayName-count-other": "یورو",
    "symbol": "€",
    "symbol-alt-narrow": "€"
  },
  "FIM": {
    "displayName": "مارکا فنلندی",
    "symbol": "FIM"
  },
  "FJD": {
    "displayName": "دولار فیجی",
    "displayName-count-zero": "دولار فیجی",
    "displayName-count-one": "دولار فیجی",
    "displayName-count-two": "دولار فیجی",
    "displayName-count-few": "دولار فیجی",
    "displayName-count-many": "دولار فیجی",
    "displayName-count-other": "دولار فیجی",
    "symbol": "FJD",
    "symbol-alt-narrow": "FJ$"
  },
  "FKP": {
    "displayName": "جنيه جزر فوکلاند",
    "displayName-count-zero": "جنيه جزر فوکلاند",
    "displayName-count-one": "جنيه جزر فوکلاند",
    "displayName-count-two": "جنيه جزر فوکلاند",
    "displayName-count-few": "جنيه جزر فوکلاند",
    "displayName-count-many": "جنيه جزر فوکلاند",
    "displayName-count-other": "جنيه جزر فوکلاند",
    "symbol": "FKP",
    "symbol-alt-narrow": "£"
  },
  "FRF": {
    "displayName": "فرنك فرنسي",

```



```

    "symbol": "FRF"
  },
  "GBP": {
    "displayName": "جنيه إسترليني",
    "displayName-count-zero": "جنيه إسترليني",
    "displayName-count-one": "جنيه إسترليني",
    "displayName-count-two": "جنيه إسترليني",
    "displayName-count-few": "جنيه إسترليني",
    "displayName-count-many": "جنيه إسترليني",
    "displayName-count-other": "جنيه إسترليني",
    "symbol": "£",
    "symbol-alt-narrow": "UK£"
  },
  "GEK": {
    "displayName": "GEK",
    "symbol": "GEK"
  },
  "GEL": {
    "displayName": "ლარი جورجى",
    "displayName-count-zero": "لاري جورجى",
    "displayName-count-one": "لاري جورجى",
    "displayName-count-two": "لاري جورجى",
    "displayName-count-few": "لاري جورجى",
    "displayName-count-many": "لاري جورجى",
    "displayName-count-other": "لاري جورجى",
    "symbol": "GEL",
    "symbol-alt-narrow": "ლ",
    "symbol-alt-variant": "ლ"
  },
  "GHC": {
    "displayName": "سيدي غاني",
    "symbol": "GHC"
  },
  "GHS": {
    "displayName": "سيدي غانا",
    "displayName-count-zero": "سيدي غانا",
    "displayName-count-one": "سيدي غانا",
    "displayName-count-two": "سيدي غانا",
    "displayName-count-few": "سيدي غانا",
    "displayName-count-many": "سيدي غانا",
    "displayName-count-other": "سيدي غانا",
    "symbol": "GHS"
  },
  "GIP": {
    "displayName": "جنيه جبل طارق",
    "displayName-count-zero": "جنيه جبل طارق",
    "displayName-count-one": "جنيه جبل طارق",
    "displayName-count-two": "جنيه جبل طارق",
    "displayName-count-few": "جنيه جبل طارق",
    "displayName-count-many": "جنيه جبل طارق",
    "displayName-count-other": "جنيه جبل طارق",
    "symbol": "GIP",
    "symbol-alt-narrow": "£"
  },
  "GMD": {
    "displayName": "دلاسي جامبي",
    "displayName-count-zero": "دلاسي جامبي",

```

```

    "displayName-count-one": "دلاسي جامبي",
    "displayName-count-two": "دلاسي جامبي",
    "displayName-count-few": "دلاسي جامبي",
    "displayName-count-many": "دلاسي جامبي",
    "displayName-count-other": "دلاسي جامبي",
    "symbol": "GMD"
  },
  "GNF": {
    "displayName": "فرنك غينيا",
    "displayName-count-zero": "فرنك غينيا",
    "displayName-count-one": "فرنك غينيا",
    "displayName-count-two": "فرنك غينيا",
    "displayName-count-few": "فرنك غينيا",
    "displayName-count-many": "فرنك غينيا",
    "displayName-count-other": "فرنك غينيا",
    "symbol": "GNF",
    "symbol-alt-narrow": "FG"
  },
  "GNS": {
    "displayName": "سيللي غينيا",
    "symbol": "GNS"
  },
  "GQE": {
    "displayName": "اكويل جونيئا غينيا الاستوائية",
    "symbol": "GQE"
  },
  "GRD": {
    "displayName": "دراخما يوناني",
    "symbol": "GRD"
  },
  "GTQ": {
    "displayName": "كوتزال جواتيمالا",
    "displayName-count-zero": "كوتزال جواتيمالا",
    "displayName-count-one": "كوتزال جواتيمالا",
    "displayName-count-two": "كوتزال جواتيمالا",
    "displayName-count-few": "كوتزال جواتيمالا",
    "displayName-count-many": "كوتزال جواتيمالا",
    "displayName-count-other": "كوتزال جواتيمالا",
    "symbol": "GTQ",
    "symbol-alt-narrow": "Q"
  },
  "GWE": {
    "displayName": "اسكود برتغالي غينيا",
    "symbol": "GWE"
  },
  "GWP": {
    "displayName": "بيزو غينيا بيساو",
    "symbol": "GWP"
  },
  "GYD": {
    "displayName": "دولار غيانا",
    "displayName-count-zero": "دولار غيانا",
    "displayName-count-one": "دولار غيانا",
    "displayName-count-two": "دولار غيانا",
    "displayName-count-few": "دولار غيانا",
    "displayName-count-many": "دولار غيانا",
    "displayName-count-other": "دولار غيانا",

```

```

    "symbol": "GYD",
    "symbol-alt-narrow": "GY$"
  },
  "HKD": {
    "displayName": "دولار هونغ كونغ",
    "displayName-count-zero": "دولار هونغ كونغ",
    "displayName-count-one": "دولار هونغ كونغ",
    "displayName-count-two": "دولار هونغ كونغ",
    "displayName-count-few": "دولار هونغ كونغ",
    "displayName-count-many": "دولار هونغ كونغ",
    "displayName-count-other": "دولار هونغ كونغ",
    "symbol": "HK$",
    "symbol-alt-narrow": "HK$"
  },
  "HNL": {
    "displayName": "ليمبيرا هندوراس",
    "displayName-count-zero": "ليمبيرا هندوراس",
    "displayName-count-one": "ليمبيرا هندوراس",
    "displayName-count-two": "ليمبيرا هندوراس",
    "displayName-count-few": "ليمبيرا هندوراس",
    "displayName-count-many": "ليمبيرا هندوراس",
    "displayName-count-other": "ليمبيرا هندوراس",
    "symbol": "HNL",
    "symbol-alt-narrow": "L"
  },
  "HRD": {
    "displayName": "دينار كرواتي",
    "symbol": "HRD"
  },
  "HRK": {
    "displayName": "كونا كرواتي",
    "displayName-count-zero": "كونا كرواتي",
    "displayName-count-one": "كونا كرواتي",
    "displayName-count-two": "كونا كرواتي",
    "displayName-count-few": "كونا كرواتي",
    "displayName-count-many": "كونا كرواتي",
    "displayName-count-other": "كونا كرواتي",
    "symbol": "HRK",
    "symbol-alt-narrow": "kn"
  },
  "HTG": {
    "displayName": "جوردي هاييتي",
    "displayName-count-zero": "جوردي هاييتي",
    "displayName-count-one": "جوردي هاييتي",
    "displayName-count-two": "جوردي هاييتي",
    "displayName-count-few": "جوردي هاييتي",
    "displayName-count-many": "جوردي هاييتي",
    "displayName-count-other": "جوردي هاييتي",
    "symbol": "HTG"
  },
  "HUF": {
    "displayName": "فورينت مجري",
    "displayName-count-zero": "فورينت مجري",
    "displayName-count-one": "فورينت مجري",
    "displayName-count-two": "فورينت مجري",
    "displayName-count-few": "فورينت مجري",
    "displayName-count-many": "فورينت مجري",

```

```

        "displayName-count-other": "فورينت مجري",
        "symbol": "HUF",
        "symbol-alt-narrow": "Ft"
    },
    "IDR": {
        "displayName": "روبية إندونيسية",
        "displayName-count-zero": "روبية إندونيسية",
        "displayName-count-one": "روبية إندونيسية",
        "displayName-count-two": "روبية إندونيسية",
        "displayName-count-few": "روبية إندونيسية",
        "displayName-count-many": "روبية إندونيسية",
        "displayName-count-other": "روبية إندونيسية",
        "symbol": "IDR",
        "symbol-alt-narrow": "Rp"
    },
    "IEP": {
        "displayName": "جنيه إيرلندي",
        "symbol": "IEP"
    },
    "ILP": {
        "displayName": "جنيه إسرائيلي",
        "symbol": "ILP"
    },
    "ILS": {
        "displayName": "شيكل إسرائيلي جديد",
        "displayName-count-zero": "شيكل إسرائيلي جديد",
        "displayName-count-one": "شيكل إسرائيلي جديد",
        "displayName-count-two": "شيكل إسرائيلي جديد",
        "displayName-count-few": "شيكل إسرائيلي جديد",
        "displayName-count-many": "شيكل إسرائيلي جديد",
        "displayName-count-other": "شيكل إسرائيلي جديد",
        "symbol": "₪",
        "symbol-alt-narrow": "₪"
    },
    "INR": {
        "displayName": "روبية هندي",
        "displayName-count-zero": "روبية هندي",
        "displayName-count-one": "روبية هندي",
        "displayName-count-two": "روبية هندي",
        "displayName-count-few": "روبية هندي",
        "displayName-count-many": "روبية هندي",
        "displayName-count-other": "روبية هندي",
        "symbol": "₹",
        "symbol-alt-narrow": "₹"
    },
    "IQD": {
        "displayName": "دينار عراقي",
        "displayName-count-zero": "دينار عراقي",
        "displayName-count-one": "دينار عراقي",
        "displayName-count-two": "دينار عراقي",
        "displayName-count-few": "دينار عراقي",
        "displayName-count-many": "دينار عراقي",
        "displayName-count-other": "دينار عراقي",
        "symbol": "د.ع."
    },
    "IRR": {
        "displayName": "ريال إيراني",

```

```

    "displayName-count-zero": "واحد إيراني",
    "displayName-count-one": "واحد إيراني",
    "displayName-count-two": "واحد إيراني",
    "displayName-count-few": "واحد إيراني",
    "displayName-count-many": "واحد إيراني",
    "displayName-count-other": "واحد إيراني",
    "symbol": "₪.ر.",
  },
  "ISK": {
    "displayName": "كرونة آيسلندي",
    "displayName-count-zero": "كرونة آيسلندي",
    "displayName-count-one": "كرونة آيسلندي",
    "displayName-count-two": "كرونة آيسلندي",
    "displayName-count-few": "كرونة آيسلندي",
    "displayName-count-many": "كرونة آيسلندي",
    "displayName-count-other": "كرونة آيسلندي",
    "symbol": "ISK",
    "symbol-alt-narrow": "kr"
  },
  "ITL": {
    "displayName": "ليرة إيطالية",
    "symbol": "ITL"
  },
  "JMD": {
    "displayName": "دولار جامايكي",
    "displayName-count-zero": "دولار جامايكي",
    "displayName-count-one": "دولار جامايكي",
    "displayName-count-two": "دولار جامايكي",
    "displayName-count-few": "دولار جامايكي",
    "displayName-count-many": "دولار جامايكي",
    "displayName-count-other": "دولار جامايكي",
    "symbol": "JMD",
    "symbol-alt-narrow": "JM$"
  },
  "JOD": {
    "displayName": "دينار أردني",
    "displayName-count-zero": "دينار أردني",
    "displayName-count-one": "دينار أردني",
    "displayName-count-two": "دينار أردني",
    "displayName-count-few": "دينار أردني",
    "displayName-count-many": "دينار أردني",
    "displayName-count-other": "دينار أردني",
    "symbol": "د.أ."
  },
  "JPY": {
    "displayName": "ين ياباني",
    "displayName-count-zero": "ين ياباني",
    "displayName-count-one": "ين ياباني",
    "displayName-count-two": "ين ياباني",
    "displayName-count-few": "ين ياباني",
    "displayName-count-many": "ين ياباني",
    "displayName-count-other": "ين ياباني",
    "symbol": "¥",
    "symbol-alt-narrow": "¥"
  },
  "KES": {
    "displayName": "شلل كيني",

```

```

    "displayName-count-zero": "شلن کینیی",
    "displayName-count-one": "شلن کینیی",
    "displayName-count-two": "شلن کینیی",
    "displayName-count-few": "شلن کینیی",
    "displayName-count-many": "شلن کینیی",
    "displayName-count-other": "شلن کینیی",
    "symbol": "KES"
  },
  "KGS": {
    "displayName": "سوم قیرغستانی",
    "displayName-count-zero": "سوم قیرغستانی",
    "displayName-count-one": "سوم قیرغستانی",
    "displayName-count-two": "سوم قیرغستانی",
    "displayName-count-few": "سوم قیرغستانی",
    "displayName-count-many": "سوم قیرغستانی",
    "displayName-count-other": "سوم قیرغستانی",
    "symbol": "KGS"
  },
  "KHR": {
    "displayName": "ریال کمبودی",
    "displayName-count-zero": "ریال کمبودی",
    "displayName-count-one": "ریال کمبودی",
    "displayName-count-two": "ریال کمبودی",
    "displayName-count-few": "ریال کمبودی",
    "displayName-count-many": "ریال کمبودی",
    "displayName-count-other": "ریال کمبودی",
    "symbol": "KHR",
    "symbol-alt-narrow": "៛"
  },
  "KMF": {
    "displayName": "فرنك جزر القمر",
    "displayName-count-zero": "فرنك جزر القمر",
    "displayName-count-one": "فرنك جزر القمر",
    "displayName-count-two": "فرنك جزر القمر",
    "displayName-count-few": "فرنك جزر القمر",
    "displayName-count-many": "فرنك جزر القمر",
    "displayName-count-other": "فرنك جزر القمر",
    "symbol": "ف.ج.ق.",
    "symbol-alt-narrow": "CF"
  },
  "KPW": {
    "displayName": "وون كوريا الشمالية",
    "displayName-count-zero": "وون كوريا الشمالية",
    "displayName-count-one": "وون كوريا الشمالية",
    "displayName-count-two": "وون كوريا الشمالية",
    "displayName-count-few": "وون كوريا الشمالية",
    "displayName-count-many": "وون كوريا الشمالية",
    "displayName-count-other": "وون كوريا الشمالية",
    "symbol": "KPW",
    "symbol-alt-narrow": "₩"
  },
  "KRH": {
    "displayName": "KRH",
    "symbol": "KRH"
  },
  "KRO": {

```

```

    "displayName": "KRO",
    "symbol": "KRO"
  },
  "KRW": {
    "displayName": "وون كوريا الجنوبية",
    "displayName-count-zero": "وون كوريا الجنوبية",
    "displayName-count-one": "وون كوريا الجنوبية",
    "displayName-count-two": "وون كوريا الجنوبية",
    "displayName-count-few": "وون كوريا الجنوبية",
    "displayName-count-many": "وون كوريا الجنوبية",
    "displayName-count-other": "وون كوريا الجنوبية",
    "symbol": "₩",
    "symbol-alt-narrow": "₩"
  },
  "KWD": {
    "displayName": "دينار كويتي",
    "displayName-count-zero": "دينار كويتي",
    "displayName-count-one": "دينار كويتي",
    "displayName-count-two": "دينار كويتي",
    "displayName-count-few": "دينار كويتي",
    "displayName-count-many": "دينار كويتي",
    "displayName-count-other": "دينار كويتي",
    "symbol": "د.ك."
  },
  "KYD": {
    "displayName": "دولار جزر كيمن",
    "displayName-count-zero": "دولار جزر كيمن",
    "displayName-count-one": "دولار جزر كيمن",
    "displayName-count-two": "دولار جزر كيمن",
    "displayName-count-few": "دولار جزر كيمن",
    "displayName-count-many": "دولار جزر كيمن",
    "displayName-count-other": "دولار جزر كيمن",
    "symbol": "KYD",
    "symbol-alt-narrow": "KY$"
  },
  "KZT": {
    "displayName": "تينغ كازاخستاني",
    "displayName-count-zero": "تينغ كازاخستاني",
    "displayName-count-one": "تينغ كازاخستاني",
    "displayName-count-two": "تينغ كازاخستاني",
    "displayName-count-few": "تينغ كازاخستاني",
    "displayName-count-many": "تينغ كازاخستاني",
    "displayName-count-other": "تينغ كازاخستاني",
    "symbol": "KZT",
    "symbol-alt-narrow": "₸"
  },
  "LAK": {
    "displayName": "كيب لاوسي",
    "displayName-count-zero": "كيب لاوسي",
    "displayName-count-one": "كيب لاوسي",
    "displayName-count-two": "كيب لاوسي",
    "displayName-count-few": "كيب لاوسي",
    "displayName-count-many": "كيب لاوسي",
    "displayName-count-other": "كيب لاوسي",
    "symbol": "LAK",
    "symbol-alt-narrow": "₭"
  },
}

```

```

"LBP": {
  "displayName": "جنيه لبناني",
  "displayName-count-zero": "جنيه لبناني",
  "displayName-count-one": "جنيه لبناني",
  "displayName-count-two": "جنيه لبناني",
  "displayName-count-few": "جنيه لبناني",
  "displayName-count-many": "جنيه لبناني",
  "displayName-count-other": "جنيه لبناني",
  "symbol": ".ل.ل",
  "symbol-alt-narrow": "L£"
},
"LKR": {
  "displayName": "روبية سريلانكية",
  "displayName-count-zero": "روبية سريلانكية",
  "displayName-count-one": "روبية سريلانكية",
  "displayName-count-two": "روبية سريلانكية",
  "displayName-count-few": "روبية سريلانكية",
  "displayName-count-many": "روبية سريلانكية",
  "displayName-count-other": "روبية سريلانكية",
  "symbol": "LKR",
  "symbol-alt-narrow": "Rs"
},
"LRD": {
  "displayName": "دولار ليبيري",
  "displayName-count-zero": "دولار ليبيري",
  "displayName-count-one": "دولار ليبيري",
  "displayName-count-two": "دولاران ليبيريان",
  "displayName-count-few": "دولارات ليبيرية",
  "displayName-count-many": "دولارًا ليبيريًا",
  "displayName-count-other": "دولار ليبيري",
  "symbol": "LRD",
  "symbol-alt-narrow": "$"
},
"LSL": {
  "displayName": "لوتي ليسوتو",
  "symbol": "LSL"
},
"LTL": {
  "displayName": "ليتة ليتوانية",
  "displayName-count-zero": "ليتة ليتوانية",
  "displayName-count-one": "ليتة ليتوانية",
  "displayName-count-two": "ليتة ليتوانية",
  "displayName-count-few": "ليتة ليتوانية",
  "displayName-count-many": "ليتة ليتوانية",
  "displayName-count-other": "ليتة ليتوانية",
  "symbol": "LTL",
  "symbol-alt-narrow": "Lt"
},
"LTT": {
  "displayName": "تالوناس ليتواني",
  "symbol": "LTT"
},
"LUC": {
  "displayName": "فرنك لوكسمبرج قابل للتحويل",
  "symbol": "LUC"
},
"LUF": {

```



```

    "displayName": "فرنك لوكسمبرج",
    "symbol": "LUF"
  },
  "LUL": {
    "displayName": "فرنك لوكسمبرج المالي",
    "symbol": "LUL"
  },
  "LVL": {
    "displayName": "لاتس لاتفيا",
    "displayName-count-zero": "لاتس لاتفي",
    "displayName-count-one": "لاتس لاتفي",
    "displayName-count-two": "لاتس لاتفي",
    "displayName-count-few": "لاتس لاتفي",
    "displayName-count-many": "لاتس لاتفي",
    "displayName-count-other": "لاتس لاتفي",
    "symbol": "LVL",
    "symbol-alt-narrow": "Ls"
  },
  "LVR": {
    "displayName": "روبل لاتفيا",
    "symbol": "LVR"
  },
  "LYD": {
    "displayName": "دينار ليبي",
    "displayName-count-zero": "دينار ليبي",
    "displayName-count-one": "دينار ليبي",
    "displayName-count-two": "ديناران ليبيان",
    "displayName-count-few": "دينارات ليبية",
    "displayName-count-many": "دينارًا ليبيا",
    "displayName-count-other": "دينار ليبي",
    "symbol": "د.ل."
  },
  "MAD": {
    "displayName": "درهم مغربي",
    "displayName-count-zero": "درهم مغربي",
    "displayName-count-one": "درهم مغربي",
    "displayName-count-two": "درهمان مغربيان",
    "displayName-count-few": "دراهم مغربية",
    "displayName-count-many": "درهمنًا مغربيًا",
    "displayName-count-other": "درهم مغربي",
    "symbol": "د.م."
  },
  "MAF": {
    "displayName": "فرنك مغربي",
    "symbol": "MAF"
  },
  "MCF": {
    "displayName": "MCF",
    "symbol": "MCF"
  },
  "MDC": {
    "displayName": "MDC",
    "symbol": "MDC"
  },
  "MDL": {
    "displayName": "ليو مولدوفي",
    "displayName-count-zero": "ليو مولدوفي",

```

```

    "displayName-count-one": "ليو مولدوفي",
    "displayName-count-two": "ليو مولدوفي",
    "displayName-count-few": "ليو مولدوفي",
    "displayName-count-many": "ليو مولدوفي",
    "displayName-count-other": "ليو مولدوفي",
    "symbol": "MDL"
  },
  "MGA": {
    "displayName": "أرياري مدغشقر",
    "displayName-count-zero": "أرياري مدغشقر",
    "displayName-count-one": "أرياري مدغشقر",
    "displayName-count-two": "أرياري مدغشقر",
    "displayName-count-few": "أرياري مدغشقر",
    "displayName-count-many": "أرياري مدغشقر",
    "displayName-count-other": "أرياري مدغشقر",
    "symbol": "MGA",
    "symbol-alt-narrow": "Ar"
  },
  "MGF": {
    "displayName": "فرنك مدغشقر",
    "symbol": "MGF"
  },
  "MKD": {
    "displayName": "دينار مقدوني",
    "displayName-count-zero": "دينار مقدوني",
    "displayName-count-one": "دينار مقدوني",
    "displayName-count-two": "ديناران مقدونيان",
    "displayName-count-few": "دينارات مقدونية",
    "displayName-count-many": "دينارًا مقدونيًا",
    "displayName-count-other": "دينار مقدوني",
    "symbol": "MKD"
  },
  "MKN": {
    "displayName": "MKN",
    "symbol": "MKN"
  },
  "MLF": {
    "displayName": "فرنك مالي",
    "symbol": "MLF"
  },
  "MMK": {
    "displayName": "كيات ميانمار",
    "displayName-count-zero": "كيات ميانمار",
    "displayName-count-one": "كيات ميانمار",
    "displayName-count-two": "كيات ميانمار",
    "displayName-count-few": "كيات ميانمار",
    "displayName-count-many": "كيات ميانمار",
    "displayName-count-other": "كيات ميانمار",
    "symbol": "MMK",
    "symbol-alt-narrow": "K"
  },
  "MNT": {
    "displayName": "توغروغ منغولي",
    "displayName-count-zero": "توغروغ منغولي",
    "displayName-count-one": "توغروغ منغولي",
    "displayName-count-two": "توغروغ منغولي",
    "displayName-count-few": "توغروغ منغولي",

```

```

    "displayName-count-many": "توغروغ منغولي",
    "displayName-count-other": "توغروغ منغولي",
    "symbol": "MNT",
    "symbol-alt-narrow": "₮"
  },
  "MOP": {
    "displayName": "باتاكا ماكاوي",
    "displayName-count-zero": "باتاكا ماكاوي",
    "displayName-count-one": "باتاكا ماكاوي",
    "displayName-count-two": "باتاكا ماكاوي",
    "displayName-count-few": "باتاكا ماكاوي",
    "displayName-count-many": "باتاكا ماكاوي",
    "displayName-count-other": "باتاكا ماكاوي",
    "symbol": "MOP"
  },
  "MRO": {
    "displayName": "أوقية موريتانية",
    "displayName-count-zero": "أوقية موريتانية",
    "displayName-count-one": "أوقية موريتانية",
    "displayName-count-two": "أوقية موريتانية",
    "displayName-count-few": "أوقية موريتانية",
    "displayName-count-many": "أوقية موريتانية",
    "displayName-count-other": "أوقية موريتانية",
    "symbol": "أ.م."
  },
  "MTL": {
    "displayName": "ليرة مالطية",
    "symbol": "MTL"
  },
  "MTP": {
    "displayName": "جنيه مالطي",
    "symbol": "MTP"
  },
  "MUR": {
    "displayName": "روبية موريشيوسية",
    "displayName-count-zero": "روبية موريشيوسية",
    "displayName-count-one": "روبية موريشيوسية",
    "displayName-count-two": "روبية موريشيوسية",
    "displayName-count-few": "روبية موريشيوسية",
    "displayName-count-many": "روبية موريشيوسية",
    "displayName-count-other": "روبية موريشيوسية",
    "symbol": "MUR",
    "symbol-alt-narrow": "Rs"
  },
  "MVR": {
    "displayName": "روفيه جزر المالديف",
    "displayName-count-zero": "روفيه جزر المالديف",
    "displayName-count-one": "روفيه جزر المالديف",
    "displayName-count-two": "روفيه جزر المالديف",
    "displayName-count-few": "روفيه جزر المالديف",
    "displayName-count-many": "روفيه جزر المالديف",
    "displayName-count-other": "روفيه جزر المالديف",
    "symbol": "MVR"
  },
  "MWK": {
    "displayName": "كواشا ملاوي",
    "displayName-count-zero": "كواشا ملاوي",

```

```

    "displayName-count-one": "كواشا مالاوي",
    "displayName-count-two": "كواشا مالاوي",
    "displayName-count-few": "كواشا مالاوي",
    "displayName-count-many": "كواشا مالاوي",
    "displayName-count-other": "كواشا مالاوي",
    "symbol": "MWK"
  },
  "MXN": {
    "displayName": "بيزو مكسيكي",
    "displayName-count-zero": "بيزو مكسيكي",
    "displayName-count-one": "بيزو مكسيكي",
    "displayName-count-two": "بيزو مكسيكي",
    "displayName-count-few": "بيزو مكسيكي",
    "displayName-count-many": "بيزو مكسيكي",
    "displayName-count-other": "بيزو مكسيكي",
    "symbol": "MX$",
    "symbol-alt-narrow": "MX$"
  },
  "MXP": {
    "displayName": "بيزو فضي مكسيكي - 1992-1861",
    "symbol": "MXP"
  },
  "MXV": {
    "displayName": "MXV",
    "symbol": "MXV"
  },
  "MYR": {
    "displayName": "رينغيت ماليزي",
    "displayName-count-zero": "رينغيت ماليزي",
    "displayName-count-one": "رينغيت ماليزي",
    "displayName-count-two": "رينغيت ماليزي",
    "displayName-count-few": "رينغيت ماليزي",
    "displayName-count-many": "رينغيت ماليزي",
    "displayName-count-other": "رينغيت ماليزي",
    "symbol": "MYR",
    "symbol-alt-narrow": "RM"
  },
  "MZE": {
    "displayName": "اسكود موزمبيقي",
    "symbol": "MZE"
  },
  "MZM": {
    "displayName": "MZM",
    "symbol": "MZM"
  },
  "MZN": {
    "displayName": "متكال موزمبيقي",
    "displayName-count-zero": "متكال موزمبيقي",
    "displayName-count-one": "متكال موزمبيقي",
    "displayName-count-two": "متكال موزمبيقي",
    "displayName-count-few": "متكال موزمبيقي",
    "displayName-count-many": "متكال موزمبيقي",
    "displayName-count-other": "متكال موزمبيقي",
    "symbol": "MZN"
  },
  "NAD": {
    "displayName": "دولار ناميبوي",

```

```

    "displayName-count-zero": "دولار نامیبي",
    "displayName-count-one": "دولار نامیبي",
    "displayName-count-two": "دولار نامیبي",
    "displayName-count-few": "دولار نامیبي",
    "displayName-count-many": "دولار نامیبي",
    "displayName-count-other": "دولار نامیبي",
    "symbol": "NAD",
    "symbol-alt-narrow": "$"
  },
  "NGN": {
    "displayName": "نایرا نیجیری",
    "displayName-count-zero": "نایرا نیجیری",
    "displayName-count-one": "نایرا نیجیری",
    "displayName-count-two": "نایرا نیجیری",
    "displayName-count-few": "نایرا نیجیری",
    "displayName-count-many": "نایرا نیجیری",
    "displayName-count-other": "نایرا نیجیری",
    "symbol": "NGN",
    "symbol-alt-narrow": "₦"
  },
  "NIC": {
    "displayName": "کوردوبه نیکاراگوا",
    "symbol": "NIC"
  },
  "NIO": {
    "displayName": "قرطبة نیکاراگوا",
    "displayName-count-zero": "قرطبة نیکاراگوا",
    "displayName-count-one": "قرطبة نیکاراگوا",
    "displayName-count-two": "قرطبة نیکاراگوا",
    "displayName-count-few": "قرطبة نیکاراگوا",
    "displayName-count-many": "قرطبة نیکاراگوا",
    "displayName-count-other": "قرطبة نیکاراگوا",
    "symbol": "NIO",
    "symbol-alt-narrow": "C$"
  },
  "NLG": {
    "displayName": "جلدر هولندي",
    "symbol": "NLG"
  },
  "NOK": {
    "displayName": "کرونة نرويجية",
    "displayName-count-zero": "کرونة نرويجية",
    "displayName-count-one": "کرونة نرويجية",
    "displayName-count-two": "کرونة نرويجية",
    "displayName-count-few": "کرونة نرويجية",
    "displayName-count-many": "کرونة نرويجية",
    "displayName-count-other": "کرونة نرويجية",
    "symbol": "NOK",
    "symbol-alt-narrow": "kr"
  },
  "NPR": {
    "displayName": "روبية نیپالي",
    "displayName-count-zero": "روبية نیپالي",
    "displayName-count-one": "روبية نیپالي",
    "displayName-count-two": "روبية نیپالي",
    "displayName-count-few": "روبية نیپالي",
    "displayName-count-many": "روبية نیپالي",

```

```

    "displayName-count-other": "روبية نيبالي",
    "symbol": "NPR",
    "symbol-alt-narrow": "Rs"
  },
  "NZD": {
    "displayName": "دولار نيوزيلندي",
    "displayName-count-zero": "دولار نيوزيلندي",
    "displayName-count-one": "دولار نيوزيلندي",
    "displayName-count-two": "دولار نيوزيلندي",
    "displayName-count-few": "دولار نيوزيلندي",
    "displayName-count-many": "دولار نيوزيلندي",
    "displayName-count-other": "دولار نيوزيلندي",
    "symbol": "NZ$",
    "symbol-alt-narrow": "NZ$"
  },
  "OMR": {
    "displayName": "رول عماني",
    "displayName-count-zero": "رول عماني",
    "displayName-count-one": "رول عماني",
    "displayName-count-two": "رول عماني",
    "displayName-count-few": "رول عماني",
    "displayName-count-many": "رول عماني",
    "displayName-count-other": "رول عماني",
    "symbol": "ر.ع."
  },
  "PAB": {
    "displayName": "بالبوا بنمي",
    "displayName-count-zero": "بالبوا بنمي",
    "displayName-count-one": "بالبوا بنمي",
    "displayName-count-two": "بالبوا بنمي",
    "displayName-count-few": "بالبوا بنمي",
    "displayName-count-many": "بالبوا بنمي",
    "displayName-count-other": "بالبوا بنمي",
    "symbol": "PAB"
  },
  "PEI": {
    "displayName": "PEI",
    "symbol": "PEI"
  },
  "PEN": {
    "displayName": "سول جديد البيرو",
    "displayName-count-zero": "سول جديد البيرو",
    "displayName-count-one": "سول جديد البيرو",
    "displayName-count-two": "سول جديد البيرو",
    "displayName-count-few": "سول جديد البيرو",
    "displayName-count-many": "سول جديد البيرو",
    "displayName-count-other": "سول جديد البيرو",
    "symbol": "PEN"
  },
  "PES": {
    "displayName": "PES",
    "symbol": "PES"
  },
  "PGK": {
    "displayName": "كينيا بابوا غينيا الجديدة",
    "displayName-count-zero": "كينيا بابوا غينيا الجديدة",
    "displayName-count-one": "كينيا بابوا غينيا الجديدة",

```

```

    "displayName-count-two": "كيننا بابوا غينيا الجديدة",
    "displayName-count-few": "كيننا بابوا غينيا الجديدة",
    "displayName-count-many": "كيننا بابوا غينيا الجديدة",
    "displayName-count-other": "كيننا بابوا غينيا الجديدة",
    "symbol": "PGK"
  },
  "PHP": {
    "displayName": "بيزو فلبيني",
    "displayName-count-zero": "بيزو فلبيني",
    "displayName-count-one": "بيزو فلبيني",
    "displayName-count-two": "بيزو فلبيني",
    "displayName-count-few": "بيزو فلبيني",
    "displayName-count-many": "بيزو فلبيني",
    "displayName-count-other": "بيزو فلبيني",
    "symbol": "PHP",
    "symbol-alt-narrow": "₱"
  },
  "PKR": {
    "displayName": "روبية باكستاني",
    "displayName-count-zero": "روبية باكستاني",
    "displayName-count-one": "روبية باكستاني",
    "displayName-count-two": "روبية باكستاني",
    "displayName-count-few": "روبية باكستاني",
    "displayName-count-many": "روبية باكستاني",
    "displayName-count-other": "روبية باكستاني",
    "symbol": "PKR",
    "symbol-alt-narrow": "Rs"
  },
  "PLN": {
    "displayName": "زلوتي بولندي",
    "displayName-count-zero": "زلوتي بولندي",
    "displayName-count-one": "زلوتي بولندي",
    "displayName-count-two": "زلوتي بولندي",
    "displayName-count-few": "زلوتي بولندي",
    "displayName-count-many": "زلوتي بولندي",
    "displayName-count-other": "زلوتي بولندي",
    "symbol": "PLN",
    "symbol-alt-narrow": "zł"
  },
  "PLZ": {
    "displayName": "زلوتي بولندي - 1950-1995",
    "symbol": "PLZ"
  },
  "PTE": {
    "displayName": "اسكود برتغالي",
    "symbol": "PTE"
  },
  "PYG": {
    "displayName": "جواراني باراجواي",
    "displayName-count-zero": "جواراني باراجواي",
    "displayName-count-one": "جواراني باراجواي",
    "displayName-count-two": "جواراني باراجواي",
    "displayName-count-few": "جواراني باراجواي",
    "displayName-count-many": "جواراني باراجواي",
    "displayName-count-other": "جواراني باراجواي",
    "symbol": "PYG",
    "symbol-alt-narrow": "₧"
  }

```

```

    },
    "QAR": {
      "displayName": "قطري",
      "displayName-count-zero": "قطري",
      "displayName-count-one": "قطري",
      "displayName-count-two": "قطري",
      "displayName-count-few": "قطري",
      "displayName-count-many": "قطري",
      "displayName-count-other": "قطري",
      "symbol": "ر.ق."
    },
    "RHD": {
      "displayName": "دولار روديسي",
      "symbol": "RHD"
    },
    "ROL": {
      "displayName": "ليو روماني قديم",
      "symbol": "ROL"
    },
    "RON": {
      "displayName": "ليو روماني",
      "displayName-count-zero": "ليو روماني",
      "displayName-count-one": "ليو روماني",
      "displayName-count-two": "ليو روماني",
      "displayName-count-few": "ليو روماني",
      "displayName-count-many": "ليو روماني",
      "displayName-count-other": "ليو روماني",
      "symbol": "RON",
      "symbol-alt-narrow": "lei"
    },
    "RSD": {
      "displayName": "دينار صربي",
      "displayName-count-zero": "دينار صربي",
      "displayName-count-one": "دينار صربي",
      "displayName-count-two": "ديناران صربيان",
      "displayName-count-few": "دينارات صربية",
      "displayName-count-many": "دينارًا صربيًا",
      "displayName-count-other": "دينار صربي",
      "symbol": "RSD"
    },
    "RUB": {
      "displayName": "روبل روسي",
      "displayName-count-zero": "روبل روسي",
      "displayName-count-one": "روبل روسي",
      "displayName-count-two": "روبل روسي",
      "displayName-count-few": "روبل روسي",
      "displayName-count-many": "روبل روسي",
      "displayName-count-other": "روبل روسي",
      "symbol": "RUB",
      "symbol-alt-narrow": "₽"
    },
    "RUR": {
      "displayName": "روبل روسي - 1998-1991",
      "symbol": "RUR",
      "symbol-alt-narrow": "p."
    },
    "RWF": {

```



```

    "displayName": "فرنك رواندي",
    "displayName-count-zero": "فرنك رواندي",
    "displayName-count-one": "فرنك رواندي",
    "displayName-count-two": "فرنك رواندي",
    "displayName-count-few": "فرنك رواندي",
    "displayName-count-many": "فرنك رواندي",
    "displayName-count-other": "فرنك رواندي",
    "symbol": "RWF",
    "symbol-alt-narrow": "RF"
  },
  "SAR": {
    "displayName": "ريال سعودي",
    "displayName-count-zero": "ريال سعودي",
    "displayName-count-one": "ريال سعودي",
    "displayName-count-two": "ريال سعودي",
    "displayName-count-few": "ريال سعودي",
    "displayName-count-many": "ريال سعودي",
    "displayName-count-other": "ريال سعودي",
    "symbol": "ر.س."
  },
  "SBD": {
    "displayName": "دولار جزر سليمان",
    "displayName-count-zero": "دولار جزر سليمان",
    "displayName-count-one": "دولار جزر سليمان",
    "displayName-count-two": "دولار جزر سليمان",
    "displayName-count-few": "دولار جزر سليمان",
    "displayName-count-many": "دولار جزر سليمان",
    "displayName-count-other": "دولار جزر سليمان",
    "symbol": "SBD",
    "symbol-alt-narrow": "SB$"
  },
  "SCR": {
    "displayName": "روبية سيشيلية",
    "displayName-count-zero": "روبية سيشيلية",
    "displayName-count-one": "روبية سيشيلية",
    "displayName-count-two": "روبية سيشيلية",
    "displayName-count-few": "روبية سيشيلية",
    "displayName-count-many": "روبية سيشيلية",
    "displayName-count-other": "روبية سيشيلية",
    "symbol": "SCR"
  },
  "SDD": {
    "displayName": "دينار سوداني",
    "symbol": "د.س."
  },
  "SDG": {
    "displayName": "جنيه سوداني",
    "displayName-count-zero": "جنيه سوداني",
    "displayName-count-one": "جنيه سوداني",
    "displayName-count-two": "جنيه سوداني",
    "displayName-count-few": "جنيهات سودانية",
    "displayName-count-many": "جنيهًا سودانيًا",
    "displayName-count-other": "جنيه سوداني",
    "symbol": "ج.س."
  },
  "SDP": {
    "displayName": "جنيه سوداني قديم",

```

```

    "symbol": "SDP"
  },
  "SEK": {
    "displayName": "كرونة سويدية",
    "displayName-count-zero": "كرونة سويدية",
    "displayName-count-one": "كرونة سويدية",
    "displayName-count-two": "كرونة سويدية",
    "displayName-count-few": "كرونة سويدية",
    "displayName-count-many": "كرونة سويدية",
    "displayName-count-other": "كرونة سويدية",
    "symbol": "SEK",
    "symbol-alt-narrow": "kr"
  },
  "SGD": {
    "displayName": "دولار سنغافوري",
    "displayName-count-zero": "دولار سنغافوري",
    "displayName-count-one": "دولار سنغافوري",
    "displayName-count-two": "دولار سنغافوري",
    "displayName-count-few": "دولار سنغافوري",
    "displayName-count-many": "دولار سنغافوري",
    "displayName-count-other": "دولار سنغافوري",
    "symbol": "SGD",
    "symbol-alt-narrow": "$"
  },
  "SHP": {
    "displayName": "جنيه سانت هيلين",
    "displayName-count-zero": "جنيه سانت هيلين",
    "displayName-count-one": "جنيه سانت هيلين",
    "displayName-count-two": "جنيه سانت هيلين",
    "displayName-count-few": "جنيه سانت هيلين",
    "displayName-count-many": "جنيه سانت هيلين",
    "displayName-count-other": "جنيه سانت هيلين",
    "symbol": "SHP",
    "symbol-alt-narrow": "£"
  },
  "SIT": {
    "displayName": "تولار سلوفيني",
    "symbol": "SIT"
  },
  "SKK": {
    "displayName": "كرونة سلوفاكية",
    "symbol": "SKK"
  },
  "SLL": {
    "displayName": "ليون سيراليوني",
    "displayName-count-zero": "ليون سيراليوني",
    "displayName-count-one": "ليون سيراليوني",
    "displayName-count-two": "ليون سيراليوني",
    "displayName-count-few": "ليون سيراليوني",
    "displayName-count-many": "ليون سيراليوني",
    "displayName-count-other": "ليون سيراليوني",
    "symbol": "SLL"
  },
  "SOS": {
    "displayName": "شلن صومالي",
    "displayName-count-zero": "شلن صومالي",
    "displayName-count-one": "شلن صومالي",

```

```

    "displayName-count-two": "شلن صومالي",
    "displayName-count-few": "شلن صومالي",
    "displayName-count-many": "شلن صومالي",
    "displayName-count-other": "شلن صومالي",
    "symbol": "SOS"
  },
  "SRD": {
    "displayName": "دولار سورينامي",
    "displayName-count-zero": "دولار سورينامي",
    "displayName-count-one": "دولار سورينامي",
    "displayName-count-two": "دولار سورينامي",
    "displayName-count-few": "دولار سورينامي",
    "displayName-count-many": "دولار سورينامي",
    "displayName-count-other": "دولار سورينامي",
    "symbol": "SRD",
    "symbol-alt-narrow": "SR$"
  },
  "SRG": {
    "displayName": "جلدر سورينامي",
    "symbol": "SRG"
  },
  "SSP": {
    "displayName": "جنيه جنوب السودان",
    "displayName-count-zero": "جنيه جنوب السودان",
    "displayName-count-one": "جنيه جنوب السودان",
    "displayName-count-two": "جنيهان جنوب السودان",
    "displayName-count-few": "جنيهات جنوب السودان",
    "displayName-count-many": "جنيهًا جنوب السودان",
    "displayName-count-other": "جنيه جنوب السودان",
    "symbol": "SSP",
    "symbol-alt-narrow": "£"
  },
  "STD": {
    "displayName": "دوبرا ساو تومي وبرينسيبي",
    "displayName-count-zero": "دوبرا ساو تومي وبرينسيبي",
    "displayName-count-one": "دوبرا ساو تومي وبرينسيبي",
    "displayName-count-two": "دوبرا ساو تومي وبرينسيبي",
    "displayName-count-few": "دوبرا ساو تومي وبرينسيبي",
    "displayName-count-many": "دوبرا ساو تومي وبرينسيبي",
    "displayName-count-other": "دوبرا ساو تومي وبرينسيبي",
    "symbol": "STD",
    "symbol-alt-narrow": "Db"
  },
  "SUR": {
    "displayName": "روبل سوفيتي",
    "symbol": "SUR"
  },
  "SVC": {
    "displayName": "كولون سلفادوري",
    "symbol": "SVC"
  },
  "SYP": {
    "displayName": "ليرة سورية",
    "displayName-count-zero": "ليرة سورية",
    "displayName-count-one": "ليرة سورية",
    "displayName-count-two": "ليرة سورية",
    "displayName-count-few": "ليرة سورية",

```

```

        "displayName-count-many": "ليرة سورية",
        "displayName-count-other": "ليرة سورية",
        "symbol": "ل.س.",
        "symbol-alt-narrow": "£"
    },
    "SZL": {
        "displayName": "ليلانجيني سوازيلندي",
        "displayName-count-zero": "ليلانجيني سوازيلندي",
        "displayName-count-one": "ليلانجيني سوازيلندي",
        "displayName-count-two": "ليلانجيني سوازيلندي",
        "displayName-count-few": "ليلانجيني سوازيلندي",
        "displayName-count-many": "ليلانجيني سوازيلندي",
        "displayName-count-other": "ليلانجيني سوازيلندي",
        "symbol": "SZL"
    },
    "THB": {
        "displayName": "باخت تايلاندي",
        "displayName-count-zero": "باخت تايلاندي",
        "displayName-count-one": "باخت تايلاندي",
        "displayName-count-two": "باخت تايلاندي",
        "displayName-count-few": "باخت تايلاندي",
        "displayName-count-many": "باخت تايلاندي",
        "displayName-count-other": "باخت تايلاندي",
        "symbol": "฿",
        "symbol-alt-narrow": "฿"
    },
    "TJR": {
        "displayName": "روبل طاجيكستاني",
        "symbol": "TJR"
    },
    "TJS": {
        "displayName": "سوموني طاجيكستاني",
        "displayName-count-zero": "سوموني طاجيكستاني",
        "displayName-count-one": "سوموني طاجيكستاني",
        "displayName-count-two": "سوموني طاجيكستاني",
        "displayName-count-few": "سوموني طاجيكستاني",
        "displayName-count-many": "سوموني طاجيكستاني",
        "displayName-count-other": "سوموني طاجيكستاني",
        "symbol": "TJS"
    },
    "TMM": {
        "displayName": "مانات تركمنستاني",
        "symbol": "TMM"
    },
    "TMT": {
        "displayName": "مانات تركمانستان",
        "displayName-count-zero": "مانات تركمانستان",
        "displayName-count-one": "مانات تركمانستان",
        "displayName-count-two": "مانات تركمانستان",
        "displayName-count-few": "مانات تركمانستان",
        "displayName-count-many": "مانات تركمانستان",
        "displayName-count-other": "مانات تركمانستان",
        "symbol": "TMT"
    },
    "TND": {
        "displayName": "دينار تونسي",
        "displayName-count-zero": "دينار تونسي",

```

```

    "displayName-count-one": "دينار تونسي",
    "displayName-count-two": "ديناران تونسيان",
    "displayName-count-few": "دينارات تونسية",
    "displayName-count-many": "دينارًا تونسيًا",
    "displayName-count-other": "دينار تونسي",
    "symbol": "د.ت.",
  },
  "TOP": {
    "displayName": "بانغا تونغ",
    "displayName-count-zero": "بانغا تونغ",
    "displayName-count-one": "بانغا تونغ",
    "displayName-count-two": "بانغا تونغ",
    "displayName-count-few": "بانغا تونغ",
    "displayName-count-many": "بانغا تونغ",
    "displayName-count-other": "بانغا تونغ",
    "symbol": "TOP",
    "symbol-alt-narrow": "T$"
  },
  "TPE": {
    "displayName": "اسكود تيموري",
    "symbol": "TPE"
  },
  "TRL": {
    "displayName": "ليرة تركي",
    "symbol": "TRL"
  },
  "TRY": {
    "displayName": "ليرة تركية",
    "displayName-count-zero": "ليرة تركية",
    "displayName-count-one": "ليرة تركية",
    "displayName-count-two": "ليرة تركية",
    "displayName-count-few": "ليرة تركية",
    "displayName-count-many": "ليرة تركية",
    "displayName-count-other": "ليرة تركية",
    "symbol": "TRY",
    "symbol-alt-narrow": "₺",
    "symbol-alt-variant": "TL"
  },
  "TTD": {
    "displayName": "دولار ترينداد وتوباغو",
    "displayName-count-zero": "دولار ترينداد وتوباغو",
    "displayName-count-one": "دولار ترينداد وتوباغو",
    "displayName-count-two": "دولار ترينداد وتوباغو",
    "displayName-count-few": "دولار ترينداد وتوباغو",
    "displayName-count-many": "دولار ترينداد وتوباغو",
    "displayName-count-other": "دولار ترينداد وتوباغو",
    "symbol": "TTD",
    "symbol-alt-narrow": "TT$"
  },
  "TWD": {
    "displayName": "دولار تايواني",
    "displayName-count-zero": "دولار تايواني",
    "displayName-count-one": "دولار تايواني",
    "displayName-count-two": "دولار تايواني",
    "displayName-count-few": "دولار تايواني",
    "displayName-count-many": "دولار تايواني",
    "displayName-count-other": "دولار تايواني",

```

```

    "symbol": "NT$",
    "symbol-alt-narrow": "NT$"
  },
  "TZS": {
    "displayName": "شلن تنزاني",
    "displayName-count-zero": "شلن تنزاني",
    "displayName-count-one": "شلن تنزاني",
    "displayName-count-two": "شلن تنزاني",
    "displayName-count-few": "شلن تنزاني",
    "displayName-count-many": "شلن تنزاني",
    "displayName-count-other": "شلن تنزاني",
    "symbol": "TZS"
  },
  "UAH": {
    "displayName": "هريفنيا أوكراني",
    "displayName-count-zero": "هريفنيا أوكراني",
    "displayName-count-one": "هريفنيا أوكراني",
    "displayName-count-two": "هريفنيا أوكراني",
    "displayName-count-few": "هريفنيا أوكراني",
    "displayName-count-many": "هريفنيا أوكراني",
    "displayName-count-other": "هريفنيا أوكراني",
    "symbol": "UAH",
    "symbol-alt-narrow": "₴"
  },
  "UAK": {
    "displayName": "UAK",
    "symbol": "UAK"
  },
  "UGS": {
    "displayName": "شلن أوغندي - 1987-1966",
    "symbol": "UGS"
  },
  "UGX": {
    "displayName": "شلن أوغندي",
    "displayName-count-zero": "شلن أوغندي",
    "displayName-count-one": "شلن أوغندي",
    "displayName-count-two": "شلن أوغندي",
    "displayName-count-few": "شلن أوغندي",
    "displayName-count-many": "شلن أوغندي",
    "displayName-count-other": "شلن أوغندي",
    "symbol": "UGX"
  },
  "USD": {
    "displayName": "دولار أمريكي",
    "displayName-count-zero": "دولار أمريكي",
    "displayName-count-one": "دولار أمريكي",
    "displayName-count-two": "دولار أمريكي",
    "displayName-count-few": "دولار أمريكي",
    "displayName-count-many": "دولار أمريكي",
    "displayName-count-other": "دولار أمريكي",
    "symbol": "US$",
    "symbol-alt-narrow": "US$"
  },
  "USN": {
    "displayName": "دولار أمريكي (اليوم التالي)",
    "symbol": "USN"
  },
}

```

```

"USS": {
  "displayName": "(دولار أمريكي) نفس اليوم",
  "symbol": "USS"
},
"UYI": {
  "displayName": "UYI",
  "symbol": "UYI"
},
"UYP": {
  "displayName": "بیزو اوروجوای - 1993-1975",
  "symbol": "UYP"
},
"UYU": {
  "displayName": "بیزو اوروجوای",
  "displayName-count-zero": "بیزو اوروجوای",
  "displayName-count-one": "بیزو اوروجوای",
  "displayName-count-two": "بیزو اوروجوای",
  "displayName-count-few": "بیزو اوروجوای",
  "displayName-count-many": "بیزو اوروجوای",
  "displayName-count-other": "بیزو اوروجوای",
  "symbol": "UYU",
  "symbol-alt-narrow": "UY$"
},
"UZS": {
  "displayName": "سوم اوزبكستاني",
  "displayName-count-zero": "سوم اوزبكستاني",
  "displayName-count-one": "سوم اوزبكستاني",
  "displayName-count-two": "سوم اوزبكستاني",
  "displayName-count-few": "سوم اوزبكستاني",
  "displayName-count-many": "سوم اوزبكستاني",
  "displayName-count-other": "سوم اوزبكستاني",
  "symbol": "UZS"
},
"VEB": {
  "displayName": "بوليفار فنزويلى - 2008-1871",
  "symbol": "VEB"
},
"VEF": {
  "displayName": "بوليفار فنزويلى",
  "displayName-count-zero": "بوليفار فنزويلى",
  "displayName-count-one": "بوليفار فنزويلى",
  "displayName-count-two": "بوليفار فنزويلى",
  "displayName-count-few": "بوليفار فنزويلى",
  "displayName-count-many": "بوليفار فنزويلى",
  "displayName-count-other": "بوليفار فنزويلى",
  "symbol": "VEF",
  "symbol-alt-narrow": "Bs"
},
"VND": {
  "displayName": "دونج فيتنامي",
  "displayName-count-zero": "دونج فيتنامي",
  "displayName-count-one": "دونج فيتنامي",
  "displayName-count-two": "دونج فيتنامي",
  "displayName-count-few": "دونج فيتنامي",
  "displayName-count-many": "دونج فيتنامي",
  "displayName-count-other": "دونج فيتنامي",
  "symbol": "₫",

```

```

    "symbol-alt-narrow": "₪"
  },
  "VNN": {
    "displayName": "VNN",
    "symbol": "VNN"
  },
  "VUV": {
    "displayName": "فاتو فانواتو",
    "displayName-count-zero": "فاتو فانواتو",
    "displayName-count-one": "فاتو فانواتو",
    "displayName-count-two": "فاتو فانواتو",
    "displayName-count-few": "فاتو فانواتو",
    "displayName-count-many": "فاتو فانواتو",
    "displayName-count-other": "فاتو فانواتو",
    "symbol": "VUV"
  },
  "WST": {
    "displayName": "تالا ساموا",
    "displayName-count-zero": "تالا ساموا",
    "displayName-count-one": "تالا ساموا",
    "displayName-count-two": "تالا ساموا",
    "displayName-count-few": "تالا ساموا",
    "displayName-count-many": "تالا ساموا",
    "displayName-count-other": "تالا ساموا",
    "symbol": "WST"
  },
  "XAF": {
    "displayName": "فرنك وسط أفريقي",
    "displayName-count-zero": "فرنك وسط أفريقي",
    "displayName-count-one": "فرنك وسط أفريقي",
    "displayName-count-two": "فرنك وسط أفريقي",
    "displayName-count-few": "فرنك وسط أفريقي",
    "displayName-count-many": "فرنك وسط أفريقي",
    "displayName-count-other": "فرنك وسط أفريقي",
    "symbol": "FCFA"
  },
  "XAG": {
    "displayName": "فضة",
    "symbol": "XAG"
  },
  "XAU": {
    "displayName": "ذهب",
    "symbol": "XAU"
  },
  "XBA": {
    "displayName": "الوحدة الأوروبية المركبة",
    "symbol": "XBA"
  },
  "XBB": {
    "displayName": "الوحدة المالية الأوروبية",
    "symbol": "XBB"
  },
  "XBC": {
    "displayName": "الوحدة الحسابية الأوروبية",
    "symbol": "XBC"
  },
  "XBD": {

```



```

    "displayName": "وحدة الحساب الأوروبية (XBD)",
    "symbol": "XBD"
  },
  "XCD": {
    "displayName": "دولار شرق الكاريبي",
    "displayName-count-zero": "دولار شرق الكاريبي",
    "displayName-count-one": "دولار شرق الكاريبي",
    "displayName-count-two": "دولار شرق الكاريبي",
    "displayName-count-few": "دولار شرق الكاريبي",
    "displayName-count-many": "دولار شرق الكاريبي",
    "displayName-count-other": "دولار شرق الكاريبي",
    "symbol": "EC$",
    "symbol-alt-narrow": "$"
  },
  "XDR": {
    "displayName": "حقوق السحب الخاصة",
    "symbol": "XDR"
  },
  "XEU": {
    "displayName": "وحدة النقد الأوروبية",
    "symbol": "XEU"
  },
  "XFO": {
    "displayName": "فرنك فرنسي ذهبي",
    "symbol": "XFO"
  },
  "XFU": {
    "displayName": "فرنك فرنسي (UIC)",
    "symbol": "XFU"
  },
  "XOF": {
    "displayName": "فرنك غرب أفريقي",
    "displayName-count-zero": "فرنك غرب أفريقي",
    "displayName-count-one": "فرنك غرب أفريقي",
    "displayName-count-two": "فرنك غرب أفريقي",
    "displayName-count-few": "فرنك غرب أفريقي",
    "displayName-count-many": "فرنك غرب أفريقي",
    "displayName-count-other": "فرنك غرب أفريقي",
    "symbol": "CFA"
  },
  "XPD": {
    "displayName": "بالاديوم",
    "symbol": "XPD"
  },
  "XPF": {
    "displayName": "فرنك سي إف بي",
    "displayName-count-zero": "فرنك سي إف بي",
    "displayName-count-one": "فرنك سي إف بي",
    "displayName-count-two": "فرنك سي إف بي",
    "displayName-count-few": "فرنك سي إف بي",
    "displayName-count-many": "فرنك سي إف بي",
    "displayName-count-other": "فرنك سي إف بي",
    "symbol": "CFPF"
  },
  "XPT": {
    "displayName": "البلاتين",
    "symbol": "XPT"
  }

```

```

    },
    "XRE": {
      "displayName": "XRE",
      "symbol": "XRE"
    },
    "XSU": {
      "displayName": "XSU",
      "symbol": "XSU"
    },
    "XTS": {
      "displayName": "كود اختبار العملة",
      "symbol": "XTS"
    },
    "XUA": {
      "displayName": "XUA",
      "symbol": "XUA"
    },
    "XXX": {
      "displayName": "عملة غير معروفة",
      "displayName-count-zero": "(عملة غير معروفة)",
      "displayName-count-one": "(عملة غير معروفة)",
      "displayName-count-two": "(عملة غير معروفة)",
      "displayName-count-few": "(عملة غير معروفة)",
      "displayName-count-many": "(عملة غير معروفة)",
      "displayName-count-other": "(عملة غير معروفة)",
      "symbol": "****"
    },
    "YDD": {
      "displayName": "دينار يمني",
      "symbol": "YDD"
    },
    "YER": {
      "displayName": "ول يمني",
      "displayName-count-zero": "ول يمني",
      "displayName-count-one": "ول يمني",
      "displayName-count-two": "ول يمني",
      "displayName-count-few": "ول يمني",
      "displayName-count-many": "ول يمني",
      "displayName-count-other": "ول يمني",
      "symbol": "ر.ي."
    },
    "YUD": {
      "displayName": "دينار يوغسلافي",
      "symbol": "YUD"
    },
    "YUM": {
      "displayName": "YUM",
      "symbol": "YUM"
    },
    "YUN": {
      "displayName": "دينار يوغسلافي قابل للتحويل",
      "symbol": "YUN"
    },
    "YUR": {
      "displayName": "YUR",
      "symbol": "YUR"
    },
  },

```

```

    "ZAL": {
      "displayName": "راند جنوب أفريقيا -مالي",
      "symbol": "ZAL"
    },
    "ZAR": {
      "displayName": "راند جنوب أفريقيا",
      "displayName-count-zero": "راند جنوب أفريقيا",
      "displayName-count-one": "راند جنوب أفريقيا",
      "displayName-count-two": "راند جنوب أفريقيا",
      "displayName-count-few": "راند جنوب أفريقيا",
      "displayName-count-many": "راند جنوب أفريقيا",
      "displayName-count-other": "راند جنوب أفريقيا",
      "symbol": "ZAR",
      "symbol-alt-narrow": "R"
    },
    "ZMK": {
      "displayName": "كواشا زامبي - 2012-1968",
      "displayName-count-zero": "كواشا زامبي - 2012-1968",
      "displayName-count-one": "كواشا زامبي - 2012-1968",
      "displayName-count-two": "كواشا زامبي - 2012-1968",
      "displayName-count-few": "كواشا زامبي - 2012-1968",
      "displayName-count-many": "كواشا زامبي - 2012-1968",
      "displayName-count-other": "كواشا زامبي - 2012-1968",
      "symbol": "ZMK"
    },
    "ZMW": {
      "displayName": "كواشا زامبي",
      "displayName-count-zero": "كواشا زامبي",
      "displayName-count-one": "كواشا زامبي",
      "displayName-count-two": "كواشا زامبي",
      "displayName-count-few": "كواشا زامبي",
      "displayName-count-many": "كواشا زامبي",
      "displayName-count-other": "كواشا زامبي",
      "symbol": "ZMW",
      "symbol-alt-narrow": "ZK"
    },
    "ZRN": {
      "displayName": "زائير زائيري جديد",
      "symbol": "ZRN"
    },
    "ZRZ": {
      "displayName": "زائير زائيري",
      "symbol": "ZRZ"
    },
    "ZWD": {
      "displayName": "دولار زمبابوي",
      "symbol": "ZWD"
    },
    "ZWL": {
      "displayName": "دولار زمبابوي 2009",
      "symbol": "ZWL"
    },
    "ZWR": {
      "displayName": "ZWR",
      "symbol": "ZWR"
    }
  }
}

```

```

    }
  }
}

```

CURRENCIES.JSX

```

{
  "main";
  {
    "ar";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 12879 $",
          "_cldrVersion";
          "30.0.3";
        }
        "language";
        "ar";
      }
      "numbers";
      {
        "currencies";
        {
          "ADP";
          {
            "displayName";
            "بيستا أندوري",
            "symbol";
            "ADP";
          }
          "AED";
          {
            "displayName";
            "درهم إماراتي",
            "displayName-count-zero";
            "درهم إماراتي",
            "displayName-count-one";
            "درهم إماراتي",
            "displayName-count-two";
            "درهم إماراتي",
            "displayName-count-few";
            "درهم إماراتي",
            "displayName-count-many";
            "درهم إماراتي",
            "displayName-count-other";
            "درهم إماراتي",
            "symbol";
            "د.إ.";
          }
          "AFA";
          {

```

```

        "displayName";
        "2002-1927 - أفغاني",
        "symbol";
        "AFA";
    }
    "AFN";
    {
        "displayName";
        "أفغاني",
        "displayName-count-zero";
        "أفغاني أفغانستان",
        "displayName-count-one";
        "أفغاني أفغانستان",
        "displayName-count-two";
        "أفغاني أفغانستان",
        "displayName-count-few";
        "أفغاني أفغانستان",
        "displayName-count-many";
        "أفغاني أفغانستان",
        "displayName-count-other";
        "أفغاني أفغانستان",
        "symbol";
        "AFN";
    }
    "ALL";
    {
        "displayName";
        "ليك ألباني",
        "displayName-count-zero";
        "ليك ألباني",
        "displayName-count-one";
        "ليك ألباني",
        "displayName-count-two";
        "ليك ألباني",
        "displayName-count-few";
        "ليك ألباني",
        "displayName-count-many";
        "ليك ألباني",
        "displayName-count-other";
        "ليك ألباني",
        "symbol";
        "ALL";
    }
    "AMD";
    {
        "displayName";
        "درام أرميني",
        "displayName-count-zero";
        "درام أرميني",
        "displayName-count-one";
        "درام أرميني",
        "displayName-count-two";
        "درام أرميني",
        "displayName-count-few";
        "درام أرميني",
        "displayName-count-many";
        "درام أرميني",
    }

```

```

        "displayName-count-other";
        "درام أرميني",
        "symbol";
        "AMD";
    }
    "ANG";
    {
        "displayName";
        "غيلدر أنتيلي هولندي",
        "displayName-count-zero";
        "غيلدر أنتيلي هولندي",
        "displayName-count-one";
        "غيلدر أنتيلي هولندي",
        "displayName-count-two";
        "غيلدر أنتيلي هولندي",
        "displayName-count-few";
        "غيلدر أنتيلي هولندي",
        "displayName-count-many";
        "غيلدر أنتيلي هولندي",
        "displayName-count-other";
        "غيلدر أنتيلي هولندي",
        "symbol";
        "ANG";
    }
    "AOA";
    {
        "displayName";
        "كوانزا أنجولي",
        "displayName-count-zero";
        "كوانزا أنجولي",
        "displayName-count-one";
        "كوانزا أنجولي",
        "displayName-count-two";
        "كوانزا أنجولي",
        "displayName-count-few";
        "كوانزا أنجولي",
        "displayName-count-many";
        "كوانزا أنجولي",
        "displayName-count-other";
        "كوانزا أنجولي",
        "symbol";
        "AOA",
        "symbol-alt-narrow";
        "Kz";
    }
    "AOK";
    {
        "displayName";
        "كوانزا أنجولي - 1990-1977",
        "symbol";
        "AOK";
    }
    "AON";
    {
        "displayName";
        "كوانزا أنجولي جديدة - 2000-1990",
        "symbol";
    }

```

```

        "AON";
    }
    "AOR";
    {
        "displayName";
        "1999 - 1995 - أنجولي معدلة",
        "symbol";
        "AOR";
    }
    "ARA";
    {
        "displayName";
        "استرال أرجنتيني",
        "symbol";
        "ARA";
    }
    "ARL";
    {
        "displayName";
        "ARL",
        "symbol";
        "ARL";
    }
    "ARM";
    {
        "displayName";
        "ARM",
        "symbol";
        "ARM";
    }
    "ARP";
    {
        "displayName";
        "1985-1983 - بيزو أرجنتيني",
        "symbol";
        "ARP";
    }
    "ARS";
    {
        "displayName";
        "بيزو أرجنتيني",
        "displayName-count-zero";
        "بيزو أرجنتيني",
        "displayName-count-one";
        "بيزو أرجنتيني",
        "displayName-count-two";
        "بيزو أرجنتيني",
        "displayName-count-few";
        "بيزو أرجنتيني",
        "displayName-count-many";
        "بيزو أرجنتيني",
        "displayName-count-other";
        "بيزو أرجنتيني",
        "symbol";
        "ARS",
        "symbol-alt-narrow";
        "AR$";
    }

```

```

    }
    "ATS";
    {
        "displayName";
        "شلن نمساوي",
        "symbol";
        "ATS";
    }
    "AUD";
    {
        "displayName";
        "دولار أسترالي",
        "displayName-count-zero";
        "دولار أسترالي",
        "displayName-count-one";
        "دولار أسترالي",
        "displayName-count-two";
        "دولار أسترالي",
        "displayName-count-few";
        "دولار أسترالي",
        "displayName-count-many";
        "دولار أسترالي",
        "displayName-count-other";
        "دولار أسترالي",
        "symbol";
        "AU$";
        "symbol-alt-narrow";
        "AU$";
    }
    "AWG";
    {
        "displayName";
        "فلورن أروبي",
        "displayName-count-zero";
        "فلورن أروبي",
        "displayName-count-one";
        "فلورن أروبي",
        "displayName-count-two";
        "فلورن أروبي",
        "displayName-count-few";
        "فلورن أروبي",
        "displayName-count-many";
        "فلورن أروبي",
        "displayName-count-other";
        "فلورن أروبي",
        "symbol";
        "AWG";
    }
    "AZM";
    {
        "displayName";
        "مانات أذربيجاني",
        "symbol";
        "AZM";
    }
    "AZN";
    {

```



```

        "displayName";
        "مانات أذربيجان",
        "displayName-count-zero";
        "مانت أذربيجاني",
        "displayName-count-one";
        "مانت أذربيجاني",
        "displayName-count-two";
        "مانت أذربيجاني",
        "displayName-count-few";
        "مانت أذربيجاني",
        "displayName-count-many";
        "مانت أذربيجاني",
        "displayName-count-other";
        "مانت أذربيجاني",
        "symbol";
        "AZN";
    }
    "BAD";
    {
        "displayName";
        "دينار البوسنة والهرسك",
        "symbol";
        "BAD";
    }
    "BAM";
    {
        "displayName";
        "مارك البوسنة والهرسك قابل للتحويل",
        "displayName-count-zero";
        "مارك البوسنة والهرسك قابل للتحويل",
        "displayName-count-one";
        "مارك البوسنة والهرسك قابل للتحويل",
        "displayName-count-two";
        "مارك البوسنة والهرسك قابل للتحويل",
        "displayName-count-few";
        "مارك البوسنة والهرسك قابل للتحويل",
        "displayName-count-many";
        "مارك البوسنة والهرسك قابل للتحويل",
        "displayName-count-other";
        "مارك البوسنة والهرسك قابل للتحويل",
        "symbol";
        "BAM",
        "symbol-alt-narrow";
        "KM";
    }
    "BAN";
    {
        "displayName";
        "BAN",
        "symbol";
        "BAN";
    }
    "BBD";
    {
        "displayName";
        "دولار بربادوسي",
        "displayName-count-zero";

```

```

        "دولار بربادوسي",
        "displayName-count-one";
        "دولار بربادوسي",
        "displayName-count-two";
        "دولار بربادوسي",
        "displayName-count-few";
        "دولار بربادوسي",
        "displayName-count-many";
        "دولار بربادوسي",
        "displayName-count-other";
        "دولار بربادوسي",
        "symbol";
        "BBD",
        "symbol-alt-narrow";
        "BB$";
    }
    "BDT";
    {
        "displayName";
        "তাকা বঙ্গলাদেশি",
        "displayName-count-zero";
        "তাকা বঙ্গলাদেশি",
        "displayName-count-one";
        "তাকা বঙ্গলাদেশি",
        "displayName-count-two";
        "তাকা বঙ্গলাদেশি",
        "displayName-count-few";
        "তাকা বঙ্গলাদেশি",
        "displayName-count-many";
        "তাকা বঙ্গলাদেশি",
        "displayName-count-other";
        "তাকা বঙ্গলাদেশি",
        "symbol";
        "BDT",
        "symbol-alt-narrow";
        "৳";
    }
    "BEC";
    {
        "displayName";
        "فرنك بلجيكي قابل للتحويل",
        "symbol";
        "BEC";
    }
    "BEF";
    {
        "displayName";
        "فرنك بلجيكي",
        "symbol";
        "BEF";
    }
    "BEL";
    {
        "displayName";
        "فرنك بلجيكي مالي",
        "symbol";
    }

```

```

        "BEL";
    }
    "BGL";
    {
        "displayName";
        "BGL",
        "symbol";
        "BGL";
    }
    "BGM";
    {
        "displayName";
        "BGM",
        "symbol";
        "BGM";
    }
    "BGN";
    {
        "displayName";
        "ليف بلغاري",
        "displayName-count-zero";
        "ليف بلغاري",
        "displayName-count-one";
        "ليف بلغاري",
        "displayName-count-two";
        "ليف بلغاري",
        "displayName-count-few";
        "ليف بلغاري",
        "displayName-count-many";
        "ليف بلغاري",
        "displayName-count-other";
        "ليف بلغاري",
        "symbol";
        "BGN";
    }
    "BGO";
    {
        "displayName";
        "BGO",
        "symbol";
        "BGO";
    }
    "BHD";
    {
        "displayName";
        "دينار بحريني",
        "displayName-count-zero";
        "دينار بحريني",
        "displayName-count-one";
        "دينار بحريني",
        "displayName-count-two";
        "دينار بحريني",
        "displayName-count-few";
        "دينار بحريني",
        "displayName-count-many";
        "دينار بحريني",
        "displayName-count-other";
    }

```

```

        "دينار بحريني",
        "symbol";
        "د.ب.";
    }
    "BIF";
    {
        "displayName";
        "فرنك بروندي",
        "displayName-count-zero";
        "فرنك بروندي",
        "displayName-count-one";
        "فرنك بروندي",
        "displayName-count-two";
        "فرنك بروندي",
        "displayName-count-few";
        "فرنك بروندي",
        "displayName-count-many";
        "فرنك بروندي",
        "displayName-count-other";
        "فرنك بروندي",
        "symbol";
        "BIF";
    }
    "BMD";
    {
        "displayName";
        "دولار برمودي",
        "displayName-count-zero";
        "دولار برمودي",
        "displayName-count-one";
        "دولار برمودي",
        "displayName-count-two";
        "دولار برمودي",
        "displayName-count-few";
        "دولار برمودي",
        "displayName-count-many";
        "دولار برمودي",
        "displayName-count-other";
        "دولار برمودي",
        "symbol";
        "BMD",
        "symbol-alt-narrow";
        "BM$";
    }
    "BND";
    {
        "displayName";
        "دولار برونائي",
        "displayName-count-zero";
        "دولار برونائي",
        "displayName-count-one";
        "دولار برونائي",
        "displayName-count-two";
        "دولار برونائي",
        "displayName-count-few";
        "دولار برونائي",
        "displayName-count-many";
    }

```

```

        "دولار برونای",
        "displayName-count-other";
        "دولار برونای",
        "symbol";
        "BND",
        "symbol-alt-narrow";
        "BN$";
    }
    "BOB";
    {
        "displayName";
        "بولیانیو بولیفی",
        "displayName-count-zero";
        "بولیانیو بولیفی",
        "displayName-count-one";
        "بولیانیو بولیفی",
        "displayName-count-two";
        "بولیانیو بولیفی",
        "displayName-count-few";
        "بولیانیو بولیفی",
        "displayName-count-many";
        "بولیانیو بولیفی",
        "displayName-count-other";
        "بولیانیو بولیفی",
        "symbol";
        "BOB",
        "symbol-alt-narrow";
        "Bs";
    }
    "BOL";
    {
        "displayName";
        "BOL",
        "symbol";
        "BOL";
    }
    "BOP";
    {
        "displayName";
        "بیزو بولیفی",
        "symbol";
        "BOP";
    }
    "BOV";
    {
        "displayName";
        "مفدول بولیفی",
        "symbol";
        "BOV";
    }
    "BRB";
    {
        "displayName";
        "نوفو کروزایرو برازیلی - 1986-1967",
        "symbol";
        "BRB";
    }
}

```

```

"BRC";
{
  "displayName";
  "کروزادو برازیلی",
  "symbol";
  "BRC";
}
"BRE";
{
  "displayName";
  "1993-1990 - کروزایرو برازیلی",
  "symbol";
  "BRE";
}
"BRL";
{
  "displayName";
  "ریال برازیلی",
  "displayName-count-zero";
  "ریال برازیلی",
  "displayName-count-one";
  "ریال برازیلی",
  "displayName-count-two";
  "ریال برازیلی",
  "displayName-count-few";
  "ریال برازیلی",
  "displayName-count-many";
  "ریال برازیلی",
  "displayName-count-other";
  "ریال برازیلی",
  "symbol";
  "R$";
  "symbol-alt-narrow";
  "R$";
}
"BRN";
{
  "displayName";
  "BRN",
  "symbol";
  "BRN";
}
"BRR";
{
  "displayName";
  "BRR",
  "symbol";
  "BRR";
}
"BRZ";
{
  "displayName";
  "BRZ",
  "symbol";
  "BRZ";
}
"BSD";

```

```

{
  "displayName";
  "دولار باهامي",
  "displayName-count-zero";
  "دولار باهامي",
  "displayName-count-one";
  "دولار باهامي",
  "displayName-count-two";
  "دولار باهامي",
  "displayName-count-few";
  "دولار باهامي",
  "displayName-count-many";
  "دولار باهامي",
  "displayName-count-other";
  "دولار باهامي",
  "symbol";
  "BSD",
  "symbol-alt-narrow";
  "BS$";
}
"BTN";
{
  "displayName";
  "نولتوم بوتاني",
  "displayName-count-zero";
  "نولتوم بوتاني",
  "displayName-count-one";
  "نولتوم بوتاني",
  "displayName-count-two";
  "نولتوم بوتاني",
  "displayName-count-few";
  "نولتوم بوتاني",
  "displayName-count-many";
  "نولتوم بوتاني",
  "displayName-count-other";
  "نولتوم بوتاني",
  "symbol";
  "BTN";
}
"BUK";
{
  "displayName";
  "کيات بورمي",
  "symbol";
  "BUK";
}
"BWP";
{
  "displayName";
  "بولا بتسواني",
  "displayName-count-zero";
  "بولا بتسواني",
  "displayName-count-one";
  "بولا بتسواني",
  "displayName-count-two";
  "بولا بتسواني",
  "displayName-count-few";
}

```

```

        "بولا بتسواني",
        "displayName-count-many";
        "بولا بتسواني",
        "displayName-count-other";
        "بولا بتسواني",
        "symbol";
        "BWP",
        "symbol-alt-narrow";
        "P";
    }
    "BYB";
    {
        "displayName";
        "1999-1994 - روبل بيلاروسي جديد",
        "symbol";
        "BYB";
    }
    "BYN";
    {
        "displayName";
        "روبل بيلاروسي",
        "displayName-count-zero";
        "روبل بيلاروسي",
        "displayName-count-one";
        "روبل بيلاروسي",
        "displayName-count-two";
        "روبل بيلاروسي",
        "displayName-count-few";
        "روبل بيلاروسي",
        "displayName-count-many";
        "روبل بيلاروسي",
        "displayName-count-other";
        "روبل بيلاروسي",
        "symbol";
        "BYN",
        "symbol-alt-narrow";
        "p.";
    }
    "BYR";
    {
        "displayName";
        "2016-2000 (روبل بيلاروسي)",
        "displayName-count-zero";
        "2016-2000 (روبل بيلاروسي)",
        "displayName-count-one";
        "2016-2000 (روبل بيلاروسي)",
        "displayName-count-two";
        "2016-2000 (روبل بيلاروسي)",
        "displayName-count-few";
        "2016-2000 (روبل بيلاروسي)",
        "displayName-count-many";
        "2016-2000 (روبل بيلاروسي)",
        "displayName-count-other";
        "2016-2000 (روبل بيلاروسي)",
        "symbol";
        "BYR";
    }
}

```



```

"BZD";
{
  "displayName";
  "دولار بليزي",
  "displayName-count-zero";
  "دولار بليزي",
  "displayName-count-one";
  "دولار بليزي",
  "displayName-count-two";
  "دولاران بليزيان",
  "displayName-count-few";
  "دولار بليزي",
  "displayName-count-many";
  "دولار بليزي",
  "displayName-count-other";
  "دولار بليزي",
  "symbol";
  "BZD",
  "symbol-alt-narrow";
  "BZ$";
}
"CAD";
{
  "displayName";
  "دولار كندي",
  "displayName-count-zero";
  "دولار كندي",
  "displayName-count-one";
  "دولار كندي",
  "displayName-count-two";
  "دولار كندي",
  "displayName-count-few";
  "دولار كندي",
  "displayName-count-many";
  "دولار كندي",
  "displayName-count-other";
  "دولار كندي",
  "symbol";
  "CA$",
  "symbol-alt-narrow";
  "CA$";
}
"CDF";
{
  "displayName";
  "فرنك كونغولي",
  "displayName-count-zero";
  "فرنك كونغولي",
  "displayName-count-one";
  "فرنك كونغولي",
  "displayName-count-two";
  "فرنك كونغولي",
  "displayName-count-few";
  "فرنك كونغولي",
  "displayName-count-many";
  "فرنك كونغولي",
  "displayName-count-other";
}

```

```

        "فرنك كونغولي",
        "symbol";
        "CDF";
    }
    "CHE";
    {
        "displayName";
        "CHE",
        "symbol";
        "CHE";
    }
    "CHF";
    {
        "displayName";
        "فرنك سويسري",
        "displayName-count-zero";
        "فرنك سويسري",
        "displayName-count-one";
        "فرنك سويسري",
        "displayName-count-two";
        "فرنك سويسري",
        "displayName-count-few";
        "فرنك سويسري",
        "displayName-count-many";
        "فرنك سويسري",
        "displayName-count-other";
        "فرنك سويسري",
        "symbol";
        "CHF";
    }
    "CHW";
    {
        "displayName";
        "CHW",
        "symbol";
        "CHW";
    }
    "CLE";
    {
        "displayName";
        "CLE",
        "symbol";
        "CLE";
    }
    "CLF";
    {
        "displayName";
        "CLF",
        "symbol";
        "CLF";
    }
    "CLP";
    {
        "displayName";
        "بيزو شيلي",
        "displayName-count-zero";
        "بيزو شيلي",

```

```

        "displayName-count-one";
        "بیزو شیلی",
        "displayName-count-two";
        "بیزو شیلی",
        "displayName-count-few";
        "بیزو شیلی",
        "displayName-count-many";
        "بیزو شیلی",
        "displayName-count-other";
        "بیزو شیلی",
        "symbol";
        "CLP",
        "symbol-alt-narrow";
        "CL$";
    }
    "CNY";
    {
        "displayName";
        "یوان صینی",
        "displayName-count-zero";
        "یوان صینی",
        "displayName-count-one";
        "یوان صینی",
        "displayName-count-two";
        "یوان صینی",
        "displayName-count-few";
        "یوان صینی",
        "displayName-count-many";
        "یوان صینی",
        "displayName-count-other";
        "یوان صینی",
        "symbol";
        "CN¥",
        "symbol-alt-narrow";
        "CN¥";
    }
    "COP";
    {
        "displayName";
        "بیزو کولومبی",
        "displayName-count-zero";
        "بیزو کولومبی",
        "displayName-count-one";
        "بیزو کولومبی",
        "displayName-count-two";
        "بیزو کولومبی",
        "displayName-count-few";
        "بیزو کولومبی",
        "displayName-count-many";
        "بیزو کولومبی",
        "displayName-count-other";
        "بیزو کولومبی",
        "symbol";
        "COP",
        "symbol-alt-narrow";
        "CO$";
    }
}

```

```

"COU";
{
  "displayName";
  "COU",
  "symbol";
  "COU";
}
"CRC";
{
  "displayName";
  "کولن کوستا ريکي",
  "displayName-count-zero";
  "کولن کوستا ريکي",
  "displayName-count-one";
  "کولن کوستا ريکي",
  "displayName-count-two";
  "کولن کوستا ريکي",
  "displayName-count-few";
  "کولن کوستا ريکي",
  "displayName-count-many";
  "کولن کوستا ريکي",
  "displayName-count-other";
  "کولن کوستا ريکي",
  "symbol";
  "CRC",
  "symbol-alt-narrow";
  "₡";
}
"CSD";
{
  "displayName";
  "دينار صربي قديم",
  "symbol";
  "CSD";
}
"CSK";
{
  "displayName";
  "کرونة تشيکوسلواکيا",
  "symbol";
  "CSK";
}
"CUC";
{
  "displayName";
  "بيزو کوبي قابل للتحويل",
  "displayName-count-zero";
  "بيزو کوبي قابل للتحويل",
  "displayName-count-one";
  "بيزو کوبي قابل للتحويل",
  "displayName-count-two";
  "بيزو کوبي قابل للتحويل",
  "displayName-count-few";
  "بيزو کوبي قابل للتحويل",
  "displayName-count-many";
  "بيزو کوبي قابل للتحويل",
  "displayName-count-other";
}

```

```

        "بيزو كوبي قابل للتحويل",
        "symbol";
    "CUC",
        "symbol-alt-narrow";
    "$";
}
"CUP";
{
    "displayName";
    "بيزو كوبي",
        "displayName-count-zero";
    "بيزو كوبي",
        "displayName-count-one";
    "بيزو كوبي",
        "displayName-count-two";
    "بيزو كوبي",
        "displayName-count-few";
    "بيزو كوبي",
        "displayName-count-many";
    "بيزو كوبي",
        "displayName-count-other";
    "بيزو كوبي",
        "symbol";
    "CUP",
        "symbol-alt-narrow";
    "CU$";
}
"CVE";
{
    "displayName";
    "اسكودو الرأس الخضراء",
        "displayName-count-zero";
    "اسكودو الرأس الخضراء",
        "displayName-count-one";
    "اسكودو الرأس الخضراء",
        "displayName-count-two";
    "اسكودو الرأس الخضراء",
        "displayName-count-few";
    "اسكودو الرأس الخضراء",
        "displayName-count-many";
    "اسكودو الرأس الخضراء",
        "displayName-count-other";
    "اسكودو الرأس الخضراء",
        "symbol";
    "CVE";
}
"CYP";
{
    "displayName";
    "جنيه قبرصي",
        "symbol";
    "CYP";
}
"CZK";
{
    "displayName";
    "كرونة تشيكية",

```

```

        "displayName-count-zero";
        "كرونة تشيكية",
        "displayName-count-one";
        "كرونة تشيكية",
        "displayName-count-two";
        "كرونة تشيكية",
        "displayName-count-few";
        "كرونة تشيكية",
        "displayName-count-many";
        "كرونة تشيكية",
        "displayName-count-other";
        "كرونة تشيكية",
        "symbol";
        "CZK",
        "symbol-alt-narrow";
        "Kč";
    }
    "DDM";
    {
        "displayName";
        "أوستمارك ألماني شرقي",
        "symbol";
        "DDM";
    }
    "DEM";
    {
        "displayName";
        "مارك ألماني",
        "symbol";
        "DEM";
    }
    "DJF";
    {
        "displayName";
        "فرنك جيبوتي",
        "displayName-count-zero";
        "فرنك جيبوتي",
        "displayName-count-one";
        "فرنك جيبوتي",
        "displayName-count-two";
        "فرنك جيبوتي",
        "displayName-count-few";
        "فرنك جيبوتي",
        "displayName-count-many";
        "فرنك جيبوتي",
        "displayName-count-other";
        "فرنك جيبوتي",
        "symbol";
        "DJF";
    }
    "DKK";
    {
        "displayName";
        "كرونة دانماركي",
        "displayName-count-zero";
        "كرونة دانماركي",
        "displayName-count-one";
    }

```

```

        "كرونة دانماركي",
        "displayName-count-two";
        "كرونة دانماركي",
        "displayName-count-few";
        "كرونة دانماركي",
        "displayName-count-many";
        "كرونة دانماركي",
        "displayName-count-other";
        "كرونة دانماركي",
        "symbol";
        "DKK",
        "symbol-alt-narrow";
        "kr";
    }
    "DOP";
    {
        "displayName";
        "بيزو الدومنيكان",
        "displayName-count-zero";
        "بيزو الدومنيكان",
        "displayName-count-one";
        "بيزو الدومنيكان",
        "displayName-count-two";
        "بيزو الدومنيكان",
        "displayName-count-few";
        "بيزو الدومنيكان",
        "displayName-count-many";
        "بيزو الدومنيكان",
        "displayName-count-other";
        "بيزو الدومنيكان",
        "symbol";
        "DOP",
        "symbol-alt-narrow";
        "DO$";
    }
    "DZD";
    {
        "displayName";
        "دينار جزائري",
        "displayName-count-zero";
        "دينار جزائري",
        "displayName-count-one";
        "دينار جزائري",
        "displayName-count-two";
        "ديناران جزائريان",
        "displayName-count-few";
        "دينارات جزائرية",
        "displayName-count-many";
        "دينارًا جزائريًا",
        "displayName-count-other";
        "دينار جزائري",
        "symbol";
        "د.ج.";
    }
    "ECS";
    {
        "displayName";

```

```

        "ECS",
        "symbol";
    "ECS";
}
"ECV";
{
    "displayName";
    "ECV",
    "symbol";
    "ECV";
}
"EEK";
{
    "displayName";
    "كرونة استونية",
    "symbol";
    "EEK";
}
"EGP";
{
    "displayName";
    "جنيه مصري",
    "displayName-count-zero";
    "جنيه مصري",
    "displayName-count-one";
    "جنيه مصري",
    "displayName-count-two";
    "جنيهان مصريان",
    "displayName-count-few";
    "جنيهات مصرية",
    "displayName-count-many";
    "جنيهاً مصرياً",
    "displayName-count-other";
    "جنيه مصري",
    "symbol";
    "ج.م.",
    "symbol-alt-narrow";
    "£";
}
"ERN";
{
    "displayName";
    "ناكفا أريتري",
    "displayName-count-zero";
    "ناكفا أريتري",
    "displayName-count-one";
    "ناكفا أريتري",
    "displayName-count-two";
    "ناكفا أريتري",
    "displayName-count-few";
    "ناكفا أريتري",
    "displayName-count-many";
    "ناكفا أريتري",
    "displayName-count-other";
    "ناكفا أريتري",
    "symbol";
    "ERN";
}

```



```

    }
    "ESA";
    {
        "displayName";
        "ESA",
        "symbol";
        "ESA";
    }
    "ESB";
    {
        "displayName";
        "ESB",
        "symbol";
        "ESB";
    }
    "ESP";
    {
        "displayName";
        "بیزیتا إسبانی",
        "symbol";
        "ESP",
        "symbol-alt-narrow";
        "₧";
    }
    "ETB";
    {
        "displayName";
        "بیر أثیوبی",
        "displayName-count-zero";
        "بیر أثیوبی",
        "displayName-count-one";
        "بیر أثیوبی",
        "displayName-count-two";
        "بیر أثیوبی",
        "displayName-count-few";
        "بیر أثیوبی",
        "displayName-count-many";
        "بیر أثیوبی",
        "displayName-count-other";
        "بیر أثیوبی",
        "symbol";
        "ETB";
    }
    "EUR";
    {
        "displayName";
        "یورو",
        "displayName-count-zero";
        "یورو",
        "displayName-count-one";
        "یورو",
        "displayName-count-two";
        "یورو",
        "displayName-count-few";
        "یورو",
        "displayName-count-many";
        "یورو",
    }

```

```

        "displayName-count-other";
        "یورو",
        "symbol";
        "€",
        "symbol-alt-narrow";
        "€";
    }
    "FIM";
    {
        "displayName";
        "مارکا فنلندي",
        "symbol";
        "FIM";
    }
    "FJD";
    {
        "displayName";
        "دولار فيجي",
        "displayName-count-zero";
        "دولار فيجي",
        "displayName-count-one";
        "دولار فيجي",
        "displayName-count-two";
        "دولار فيجي",
        "displayName-count-few";
        "دولار فيجي",
        "displayName-count-many";
        "دولار فيجي",
        "displayName-count-other";
        "دولار فيجي",
        "symbol";
        "FJD",
        "symbol-alt-narrow";
        "FJ$";
    }
    "FKP";
    {
        "displayName";
        "جنيه جزر فوكلاند",
        "displayName-count-zero";
        "جنيه جزر فوكلاند",
        "displayName-count-one";
        "جنيه جزر فوكلاند",
        "displayName-count-two";
        "جنيه جزر فوكلاند",
        "displayName-count-few";
        "جنيه جزر فوكلاند",
        "displayName-count-many";
        "جنيه جزر فوكلاند",
        "displayName-count-other";
        "جنيه جزر فوكلاند",
        "symbol";
        "FKP",
        "symbol-alt-narrow";
        "£";
    }
    "FRF";

```

```

    {
      "displayName";
      "فرنك فرنسي",
      "symbol";
      "FRF";
    }
    "GBP";
    {
      "displayName";
      "جنيه إسترليني",
      "displayName-count-zero";
      "جنيه إسترليني",
      "displayName-count-one";
      "جنيه إسترليني",
      "displayName-count-two";
      "جنيه إسترليني",
      "displayName-count-few";
      "جنيه إسترليني",
      "displayName-count-many";
      "جنيه إسترليني",
      "displayName-count-other";
      "جنيه إسترليني",
      "symbol";
      "£",
      "symbol-alt-narrow";
      "UK£";
    }
    "GEK";
    {
      "displayName";
      "GEK",
      "symbol";
      "GEK";
    }
    "GEL";
    {
      "displayName";
      "ლარი جورج",
      "displayName-count-zero";
      "ლარი جورج",
      "displayName-count-one";
      "ლარი جورج",
      "displayName-count-two";
      "ლარი جورج",
      "displayName-count-few";
      "ლარი جورج",
      "displayName-count-many";
      "ლარი جورج",
      "displayName-count-other";
      "ლარი جورج",
      "symbol";
      "GEL",
      "symbol-alt-narrow";
      "ლ",
      "symbol-alt-variant";
      "ლ";
    }
  }

```

```

"GHC";
{
  "displayName";
  "سيدي غاني",
  "symbol";
  "GHC";
}
"GHS";
{
  "displayName";
  "سيدي غانا",
  "displayName-count-zero";
  "سيدي غانا",
  "displayName-count-one";
  "سيدي غانا",
  "displayName-count-two";
  "سيدي غانا",
  "displayName-count-few";
  "سيدي غانا",
  "displayName-count-many";
  "سيدي غانا",
  "displayName-count-other";
  "سيدي غانا",
  "symbol";
  "GHS";
}
"GIP";
{
  "displayName";
  "جنيه جبل طارق",
  "displayName-count-zero";
  "جنيه جبل طارق",
  "displayName-count-one";
  "جنيه جبل طارق",
  "displayName-count-two";
  "جنيه جبل طارق",
  "displayName-count-few";
  "جنيه جبل طارق",
  "displayName-count-many";
  "جنيه جبل طارق",
  "displayName-count-other";
  "جنيه جبل طارق",
  "symbol";
  "GIP",
  "symbol-alt-narrow";
  "£";
}
"GMD";
{
  "displayName";
  "دلاسي جامبي",
  "displayName-count-zero";
  "دلاسي جامبي",
  "displayName-count-one";
  "دلاسي جامبي",
  "displayName-count-two";
  "دلاسي جامبي",

```

```

        "displayName-count-few";
        "دلاسي جامبي",
        "displayName-count-many";
        "دلاسي جامبي",
        "displayName-count-other";
        "دلاسي جامبي",
        "symbol";
        "GMD";
    }
    "GNF";
    {
        "displayName";
        "فرنك غينيا",
        "displayName-count-zero";
        "فرنك غينيا",
        "displayName-count-one";
        "فرنك غينيا",
        "displayName-count-two";
        "فرنك غينيا",
        "displayName-count-few";
        "فرنك غينيا",
        "displayName-count-many";
        "فرنك غينيا",
        "displayName-count-other";
        "فرنك غينيا",
        "symbol";
        "GNF";
        "symbol-alt-narrow";
        "FG";
    }
    "GNS";
    {
        "displayName";
        "سيلي غينيا",
        "symbol";
        "GNS";
    }
    "GQE";
    {
        "displayName";
        "اكويل جونيلا غينيا الاستوائية",
        "symbol";
        "GQE";
    }
    "GRD";
    {
        "displayName";
        "دراخما يوناني",
        "symbol";
        "GRD";
    }
    "GTQ";
    {
        "displayName";
        "كوتزال جواتيمالا",
        "displayName-count-zero";
        "كوتزال جواتيمالا",

```

```

        "displayName-count-one";
        "کوئز ال جواتیمالا",
        "displayName-count-two";
        "کوئز ال جواتیمالا",
        "displayName-count-few";
        "کوئز ال جواتیمالا",
        "displayName-count-many";
        "کوئز ال جواتیمالا",
        "displayName-count-other";
        "کوئز ال جواتیمالا",
        "symbol";
        "GTQ",
        "symbol-alt-narrow";
        "Q";
    }
    "GWE";
    {
        "displayName";
        "اسکود برتغالی غینیا",
        "symbol";
        "GWE";
    }
    "GWP";
    {
        "displayName";
        "بیزو غینیا بیساو",
        "symbol";
        "GWP";
    }
    "GYD";
    {
        "displayName";
        "دولار غیانا",
        "displayName-count-zero";
        "دولار غیانا",
        "displayName-count-one";
        "دولار غیانا",
        "displayName-count-two";
        "دولار غیانا",
        "displayName-count-few";
        "دولار غیانا",
        "displayName-count-many";
        "دولار غیانا",
        "displayName-count-other";
        "دولار غیانا",
        "symbol";
        "GYD",
        "symbol-alt-narrow";
        "GY$";
    }
    "HKD";
    {
        "displayName";
        "دولار هونگ کونگ",
        "displayName-count-zero";
        "دولار هونگ کونگ",
        "displayName-count-one";
    }

```

```

        "دولار هونغ كونغ",
        "displayName-count-two";
        "دولار هونغ كونغ",
        "displayName-count-few";
        "دولار هونغ كونغ",
        "displayName-count-many";
        "دولار هونغ كونغ",
        "displayName-count-other";
        "دولار هونغ كونغ",
        "symbol";
        "HK$",
        "symbol-alt-narrow";
        "HK$";
    }
    "HNL";
    {
        "displayName";
        "ليمبيرا هندوراس",
        "displayName-count-zero";
        "ليمبيرا هندوراس",
        "displayName-count-one";
        "ليمبيرا هندوراس",
        "displayName-count-two";
        "ليمبيرا هندوراس",
        "displayName-count-few";
        "ليمبيرا هندوراس",
        "displayName-count-many";
        "ليمبيرا هندوراس",
        "displayName-count-other";
        "ليمبيرا هندوراس",
        "symbol";
        "HNL",
        "symbol-alt-narrow";
        "L";
    }
    "HRD";
    {
        "displayName";
        "دينار كرواتي",
        "symbol";
        "HRD";
    }
    "HRK";
    {
        "displayName";
        "كونا كرواتي",
        "displayName-count-zero";
        "كونا كرواتي",
        "displayName-count-one";
        "كونا كرواتي",
        "displayName-count-two";
        "كونا كرواتي",
        "displayName-count-few";
        "كونا كرواتي",
        "displayName-count-many";
        "كونا كرواتي",
        "displayName-count-other";
    }

```

```

        "کونا کرواتى",
        "symbol";
    "HRK",
        "symbol-alt-narrow";
    "kn";
}
"HTG";
{
    "displayName";
    "جوردی هایتي",
        "displayName-count-zero";
    "جوردی هایتي",
        "displayName-count-one";
    "جوردی هایتي",
        "displayName-count-two";
    "جوردی هایتي",
        "displayName-count-few";
    "جوردی هایتي",
        "displayName-count-many";
    "جوردی هایتي",
        "displayName-count-other";
    "جوردی هایتي",
        "symbol";
    "HTG";
}
"HUF";
{
    "displayName";
    "فورينت مجري",
        "displayName-count-zero";
    "فورينت مجري",
        "displayName-count-one";
    "فورينت مجري",
        "displayName-count-two";
    "فورينت مجري",
        "displayName-count-few";
    "فورينت مجري",
        "displayName-count-many";
    "فورينت مجري",
        "displayName-count-other";
    "فورينت مجري",
        "symbol";
    "HUF",
        "symbol-alt-narrow";
    "Ft";
}
"IDR";
{
    "displayName";
    "روبية إندونيسية",
        "displayName-count-zero";
    "روبية إندونيسية",
        "displayName-count-one";
    "روبية إندونيسية",
        "displayName-count-two";
    "روبية إندونيسية",
        "displayName-count-few";
}

```



```

        "روبية إندونيسية",
        "displayName-count-many";
        "روبية إندونيسية",
        "displayName-count-other";
        "روبية إندونيسية",
        "symbol";
        "IDR",
        "symbol-alt-narrow";
        "Rp";
    }
    "IEP";
    {
        "displayName";
        "جنيه إيرلندي",
        "symbol";
        "IEP";
    }
    "ILP";
    {
        "displayName";
        "جنيه إسرائيلي",
        "symbol";
        "ILP";
    }
    "ILS";
    {
        "displayName";
        "شيكل إسرائيلي جديد",
        "displayName-count-zero";
        "شيكل إسرائيلي جديد",
        "displayName-count-one";
        "شيكل إسرائيلي جديد",
        "displayName-count-two";
        "شيكل إسرائيلي جديد",
        "displayName-count-few";
        "شيكل إسرائيلي جديد",
        "displayName-count-many";
        "شيكل إسرائيلي جديد",
        "displayName-count-other";
        "شيكل إسرائيلي جديد",
        "symbol";
        "₪",
        "symbol-alt-narrow";
        "₪";
    }
    "INR";
    {
        "displayName";
        "روبية هندي",
        "displayName-count-zero";
        "روبية هندي",
        "displayName-count-one";
        "روبية هندي",
        "displayName-count-two";
        "روبية هندي",
        "displayName-count-few";
        "روبية هندي",
    }

```

```

        "displayName-count-many";
        "روبية هندي",
        "displayName-count-other";
        "روبية هندي",
        "symbol";
        "₹",
        "symbol-alt-narrow";
        "₹";
    }
    "IQD";
    {
        "displayName";
        "دينار عراقي",
        "displayName-count-zero";
        "دينار عراقي",
        "displayName-count-one";
        "دينار عراقي",
        "displayName-count-two";
        "دينار عراقي",
        "displayName-count-few";
        "دينار عراقي",
        "displayName-count-many";
        "دينار عراقي",
        "displayName-count-other";
        "دينار عراقي",
        "symbol";
        "د.ع.";
    }
    "IRR";
    {
        "displayName";
        "ریال ایرانی",
        "displayName-count-zero";
        "ریال ایرانی",
        "displayName-count-one";
        "ریال ایرانی",
        "displayName-count-two";
        "ریال ایرانی",
        "displayName-count-few";
        "ریال ایرانی",
        "displayName-count-many";
        "ریال ایرانی",
        "displayName-count-other";
        "ریال ایرانی",
        "symbol";
        "ر.ا.";
    }
    "ISK";
    {
        "displayName";
        "کرونة ايسلندي",
        "displayName-count-zero";
        "کرونة ايسلندي",
        "displayName-count-one";
        "کرونة ايسلندي",
        "displayName-count-two";
        "کرونة ايسلندي",
    }

```

```

        "displayName-count-few";
        "كرونه آيسلندي",
        "displayName-count-many";
        "كرونه آيسلندي",
        "displayName-count-other";
        "كرونه آيسلندي",
        "symbol";
        "ISK",
        "symbol-alt-narrow";
        "kr";
    }
    "ITL";
    {
        "displayName";
        "ليرة إيطالية",
        "symbol";
        "ITL";
    }
    "JMD";
    {
        "displayName";
        "دولار جامايكي",
        "displayName-count-zero";
        "دولار جامايكي",
        "displayName-count-one";
        "دولار جامايكي",
        "displayName-count-two";
        "دولار جامايكي",
        "displayName-count-few";
        "دولار جامايكي",
        "displayName-count-many";
        "دولار جامايكي",
        "displayName-count-other";
        "دولار جامايكي",
        "symbol";
        "JMD",
        "symbol-alt-narrow";
        "JM$";
    }
    "JOD";
    {
        "displayName";
        "دينار أردني",
        "displayName-count-zero";
        "دينار أردني",
        "displayName-count-one";
        "دينار أردني",
        "displayName-count-two";
        "دينار أردني",
        "displayName-count-few";
        "دينار أردني",
        "displayName-count-many";
        "دينار أردني",
        "displayName-count-other";
        "دينار أردني",
        "symbol";
        "د.أ.";
    }

```

```

    }
    "JPY";
    {
        "displayName";
        "ين ياباني",
        "displayName-count-zero";
        "ين ياباني",
        "displayName-count-one";
        "ين ياباني",
        "displayName-count-two";
        "ين ياباني",
        "displayName-count-few";
        "ين ياباني",
        "displayName-count-many";
        "ين ياباني",
        "displayName-count-other";
        "ين ياباني",
        "symbol";
        "JP¥",
        "symbol-alt-narrow";
        "JP¥";
    }
    "KES";
    {
        "displayName";
        "شلن كينيي",
        "displayName-count-zero";
        "شلن كينيي",
        "displayName-count-one";
        "شلن كينيي",
        "displayName-count-two";
        "شلن كينيي",
        "displayName-count-few";
        "شلن كينيي",
        "displayName-count-many";
        "شلن كينيي",
        "displayName-count-other";
        "شلن كينيي",
        "symbol";
        "KES";
    }
    "KGS";
    {
        "displayName";
        "سوم قيرغستاني",
        "displayName-count-zero";
        "سوم قيرغستاني",
        "displayName-count-one";
        "سوم قيرغستاني",
        "displayName-count-two";
        "سوم قيرغستاني",
        "displayName-count-few";
        "سوم قيرغستاني",
        "displayName-count-many";
        "سوم قيرغستاني",
        "displayName-count-other";
        "سوم قيرغستاني",
    }

```

```

        "symbol";
        "KGS";
    }
    "KHR";
    {
        "displayName";
        "ریال کمبودی",
        "displayName-count-zero";
        "ریال کمبودی",
        "displayName-count-one";
        "ریال کمبودی",
        "displayName-count-two";
        "ریال کمبودی",
        "displayName-count-few";
        "ریال کمبودی",
        "displayName-count-many";
        "ریال کمبودی",
        "displayName-count-other";
        "ریال کمبودی",
        "symbol";
        "KHR",
        "symbol-alt-narrow";
        "₭";
    }
    "KMF";
    {
        "displayName";
        "فرنك جزر القمر",
        "displayName-count-zero";
        "فرنك جزر القمر",
        "displayName-count-one";
        "فرنك جزر القمر",
        "displayName-count-two";
        "فرنك جزر القمر",
        "displayName-count-few";
        "فرنك جزر القمر",
        "displayName-count-many";
        "فرنك جزر القمر",
        "displayName-count-other";
        "فرنك جزر القمر",
        "symbol";
        "ف.ج.ق.",
        "symbol-alt-narrow";
        "CF";
    }
    "KPW";
    {
        "displayName";
        "وون كوريا الشمالية",
        "displayName-count-zero";
        "وون كوريا الشمالية",
        "displayName-count-one";
        "وون كوريا الشمالية",
        "displayName-count-two";
        "وون كوريا الشمالية",
        "displayName-count-few";
    }

```

```

        "وون كوريا الشمالية",
        "displayName-count-many";
        "وون كوريا الشمالية",
        "displayName-count-other";
        "وون كوريا الشمالية",
        "symbol";
        "KPW",
        "symbol-alt-narrow";
        "₩";
    }
    "KRH";
    {
        "displayName";
        "KRH",
        "symbol";
        "KRH";
    }
    "KRO";
    {
        "displayName";
        "KRO",
        "symbol";
        "KRO";
    }
    "KRW";
    {
        "displayName";
        "وون كوريا الجنوبية",
        "displayName-count-zero";
        "وون كوريا الجنوبية",
        "displayName-count-one";
        "وون كوريا الجنوبية",
        "displayName-count-two";
        "وون كوريا الجنوبية",
        "displayName-count-few";
        "وون كوريا الجنوبية",
        "displayName-count-many";
        "وون كوريا الجنوبية",
        "displayName-count-other";
        "وون كوريا الجنوبية",
        "symbol";
        "₩",
        "symbol-alt-narrow";
        "₩";
    }
    "KWD";
    {
        "displayName";
        "دينار كويتي",
        "displayName-count-zero";
        "دينار كويتي",
        "displayName-count-one";
        "دينار كويتي",
        "displayName-count-two";
        "دينار كويتي",
        "displayName-count-few";
        "دينار كويتي",
    }

```

```

        "displayName-count-many";
        "دينار كويتي",
        "displayName-count-other";
        "دينار كويتي",
        "symbol";
        "د.ك.";
    }
    "KYD";
    {
        "displayName";
        "دولار جزر كيمن",
        "displayName-count-zero";
        "دولار جزر كيمن",
        "displayName-count-one";
        "دولار جزر كيمن",
        "displayName-count-two";
        "دولار جزر كيمن",
        "displayName-count-few";
        "دولار جزر كيمن",
        "displayName-count-many";
        "دولار جزر كيمن",
        "displayName-count-other";
        "دولار جزر كيمن",
        "symbol";
        "KYD",
        "symbol-alt-narrow";
        "KY$";
    }
    "KZT";
    {
        "displayName";
        "تينغ كازاخستاني",
        "displayName-count-zero";
        "تينغ كازاخستاني",
        "displayName-count-one";
        "تينغ كازاخستاني",
        "displayName-count-two";
        "تينغ كازاخستاني",
        "displayName-count-few";
        "تينغ كازاخستاني",
        "displayName-count-many";
        "تينغ كازاخستاني",
        "displayName-count-other";
        "تينغ كازاخستاني",
        "symbol";
        "KZT",
        "symbol-alt-narrow";
        "Т";
    }
    "LAK";
    {
        "displayName";
        "كيب لاوسي",
        "displayName-count-zero";
        "كيب لاوسي",
        "displayName-count-one";
        "كيب لاوسي",
    }

```

```

        "displayName-count-two";
        "ກີບ ລາວ",
        "displayName-count-few";
        "ກີບ ລາວ",
        "displayName-count-many";
        "ກີບ ລາວ",
        "displayName-count-other";
        "ກີບ ລາວ",
        "symbol";
        "LAK",
        "symbol-alt-narrow";
        "₭";
    }
    "LBP";
    {
        "displayName";
        "جنيه لبناني",
        "displayName-count-zero";
        "جنيه لبناني",
        "displayName-count-one";
        "جنيه لبناني",
        "displayName-count-two";
        "جنيه لبناني",
        "displayName-count-few";
        "جنيه لبناني",
        "displayName-count-many";
        "جنيه لبناني",
        "displayName-count-other";
        "جنيه لبناني",
        "symbol";
        "ل.ل.",
        "symbol-alt-narrow";
        "L£";
    }
    "LKR";
    {
        "displayName";
        "රුපියල් ස්‍රී ලාංකික",
        "displayName-count-zero";
        "රුපියල් ස්‍රී ලාංකික",
        "displayName-count-one";
        "රුපියල් ස්‍රී ලාංකික",
        "displayName-count-two";
        "රුපියල් ස්‍රී ලාංකික",
        "displayName-count-few";
        "රුපියල් ස්‍රී ලාංකික",
        "displayName-count-many";
        "රුපියල් ස්‍රී ලාංකික",
        "displayName-count-other";
        "රුපියල් ස්‍රී ලාංකික",
        "symbol";
        "LKR",
        "symbol-alt-narrow";
        "Rs";
    }
    "LRD";
    {

```



```

        "displayName";
        "دولار ليبيري",
        "displayName-count-zero";
        "دولار ليبيري",
        "displayName-count-one";
        "دولار ليبيري",
        "displayName-count-two";
        "دولاران ليبيريان",
        "displayName-count-few";
        "دولارات ليبيرية",
        "displayName-count-many";
        "دولارًا ليبيريًا",
        "displayName-count-other";
        "دولار ليبيري",
        "symbol";
        "LRD",
        "symbol-alt-narrow";
        "$";
    }
    "LSL";
    {
        "displayName";
        "لوتي ليسوتو",
        "symbol";
        "LSL";
    }
    "LTL";
    {
        "displayName";
        "ليتلا ليتوانية",
        "displayName-count-zero";
        "ليتلا ليتوانية",
        "displayName-count-one";
        "ليتلا ليتوانية",
        "displayName-count-two";
        "ليتلا ليتوانية",
        "displayName-count-few";
        "ليتلا ليتوانية",
        "displayName-count-many";
        "ليتلا ليتوانية",
        "displayName-count-other";
        "ليتلا ليتوانية",
        "symbol";
        "LTL",
        "symbol-alt-narrow";
        "Lt";
    }
    "LTT";
    {
        "displayName";
        "تالوناس ليتواني",
        "symbol";
        "LTT";
    }
    "LUC";
    {
        "displayName";

```

```

        "فرنك لوكسمبرج قابل للتحويل",
        "symbol";
        "LUC";
    }
    "LUF";
    {
        "displayName";
        "فرنك لوكسمبرج",
        "symbol";
        "LUF";
    }
    "LUL";
    {
        "displayName";
        "فرنك لوكسمبرج المالي",
        "symbol";
        "LUL";
    }
    "LVL";
    {
        "displayName";
        "لاتس لاتفيا",
        "displayName-count-zero";
        "لاتس لاتفي",
        "displayName-count-one";
        "لاتس لاتفي",
        "displayName-count-two";
        "لاتس لاتفي",
        "displayName-count-few";
        "لاتس لاتفي",
        "displayName-count-many";
        "لاتس لاتفي",
        "displayName-count-other";
        "لاتس لاتفي",
        "symbol";
        "LVL",
        "symbol-alt-narrow";
        "Ls";
    }
    "LVR";
    {
        "displayName";
        "روبل لاتفيا",
        "symbol";
        "LVR";
    }
    "LYD";
    {
        "displayName";
        "دينار ليبي",
        "displayName-count-zero";
        "دينار ليبي",
        "displayName-count-one";
        "دينار ليبي",
        "displayName-count-two";
        "ديناران ليبيان",
        "displayName-count-few";
    }

```

```

        "دينارات ليبية",
        "displayName-count-many";
        "دينارًا ليبيا",
        "displayName-count-other";
        "دينار ليبي",
        "symbol";
        "د.ل.";
    }
    "MAD";
    {
        "displayName";
        "درهم مغربي",
        "displayName-count-zero";
        "درهم مغربي",
        "displayName-count-one";
        "درهم مغربي",
        "displayName-count-two";
        "درهمان مغربيان",
        "displayName-count-few";
        "دراهم مغربية",
        "displayName-count-many";
        "درهما مغربيًا",
        "displayName-count-other";
        "درهم مغربي",
        "symbol";
        "د.م.";
    }
    "MAF";
    {
        "displayName";
        "فرنك مغربي",
        "symbol";
        "MAF";
    }
    "MCF";
    {
        "displayName";
        "MCF",
        "symbol";
        "MCF";
    }
    "MDC";
    {
        "displayName";
        "MDC",
        "symbol";
        "MDC";
    }
    "MDL";
    {
        "displayName";
        "ليو مولدوفي",
        "displayName-count-zero";
        "ليو مولدوفي",
        "displayName-count-one";
        "ليو مولدوفي",
        "displayName-count-two";
    }

```

```

        "ليو مولدوفي",
        "displayName-count-few";
        "ليو مولدوفي",
        "displayName-count-many";
        "ليو مولدوفي",
        "displayName-count-other";
        "ليو مولدوفي",
        "symbol";
        "MDL";
    }
    "MGA";
    {
        "displayName";
        "أرياري مدغشقر",
        "displayName-count-zero";
        "أرياري مدغشقر",
        "displayName-count-one";
        "أرياري مدغشقر",
        "displayName-count-two";
        "أرياري مدغشقر",
        "displayName-count-few";
        "أرياري مدغشقر",
        "displayName-count-many";
        "أرياري مدغشقر",
        "displayName-count-other";
        "أرياري مدغشقر",
        "symbol";
        "MGA",
        "symbol-alt-narrow";
        "Ar";
    }
    "MGF";
    {
        "displayName";
        "فرنك مدغشقر",
        "symbol";
        "MGF";
    }
    "MKD";
    {
        "displayName";
        "دينار مقدوني",
        "displayName-count-zero";
        "دينار مقدوني",
        "displayName-count-one";
        "دينار مقدوني",
        "displayName-count-two";
        "ديناران مقدونيان",
        "displayName-count-few";
        "دينارات مقدونية",
        "displayName-count-many";
        "دينارًا مقدونيًا",
        "displayName-count-other";
        "دينار مقدوني",
        "symbol";
        "MKD";
    }
}

```

```

"MKN";
{
  "displayName";
  "MKN",
  "symbol";
  "MKN";
}
"MLF";
{
  "displayName";
  "فرنك مالي",
  "symbol";
  "MLF";
}
"MMK";
{
  "displayName";
  "كيات ميانمار",
  "displayName-count-zero";
  "كيات ميانمار",
  "displayName-count-one";
  "كيات ميانمار",
  "displayName-count-two";
  "كيات ميانمار",
  "displayName-count-few";
  "كيات ميانمار",
  "displayName-count-many";
  "كيات ميانمار",
  "displayName-count-other";
  "كيات ميانمار",
  "symbol";
  "MMK",
  "symbol-alt-narrow";
  "K";
}
"MNT";
{
  "displayName";
  "توغروغ منغولي",
  "displayName-count-zero";
  "توغروغ منغولي",
  "displayName-count-one";
  "توغروغ منغولي",
  "displayName-count-two";
  "توغروغ منغولي",
  "displayName-count-few";
  "توغروغ منغولي",
  "displayName-count-many";
  "توغروغ منغولي",
  "displayName-count-other";
  "توغروغ منغولي",
  "symbol";
  "MNT",
  "symbol-alt-narrow";
  "₮";
}
"MOP";

```

```

    {
      "displayName";
      "باتاكا ماكاوي",
      "displayName-count-zero";
      "باتاكا ماكاوي",
      "displayName-count-one";
      "باتاكا ماكاوي",
      "displayName-count-two";
      "باتاكا ماكاوي",
      "displayName-count-few";
      "باتاكا ماكاوي",
      "displayName-count-many";
      "باتاكا ماكاوي",
      "displayName-count-other";
      "باتاكا ماكاوي",
      "symbol";
      "MOP";
    }
    "MRO";
    {
      "displayName";
      "أوقية موريتانية",
      "displayName-count-zero";
      "أوقية موريتانية",
      "displayName-count-one";
      "أوقية موريتانية",
      "displayName-count-two";
      "أوقية موريتانية",
      "displayName-count-few";
      "أوقية موريتانية",
      "displayName-count-many";
      "أوقية موريتانية",
      "displayName-count-other";
      "أوقية موريتانية",
      "symbol";
      "أ.م.";
    }
    "MTL";
    {
      "displayName";
      "ليرة مالطية",
      "symbol";
      "MTL";
    }
    "MTP";
    {
      "displayName";
      "جنيه مالطي",
      "symbol";
      "MTP";
    }
    "MUR";
    {
      "displayName";
      "روبية موريشيوسية",
      "displayName-count-zero";
      "روبية موريشيوسية",

```

```

        "displayName-count-one";
        "روبية موريشيوسية",
        "displayName-count-two";
        "روبية موريشيوسية",
        "displayName-count-few";
        "روبية موريشيوسية",
        "displayName-count-many";
        "روبية موريشيوسية",
        "displayName-count-other";
        "روبية موريشيوسية",
        "symbol";
        "MUR",
        "symbol-alt-narrow";
        "Rs";
    }
    "MVR";
    {
        "displayName";
        "روفيه جزر المالديف",
        "displayName-count-zero";
        "روفيه جزر المالديف",
        "displayName-count-one";
        "روفيه جزر المالديف",
        "displayName-count-two";
        "روفيه جزر المالديف",
        "displayName-count-few";
        "روفيه جزر المالديف",
        "displayName-count-many";
        "روفيه جزر المالديف",
        "displayName-count-other";
        "روفيه جزر المالديف",
        "symbol";
        "MVR";
    }
    "MWK";
    {
        "displayName";
        "كواشا ملاوي",
        "displayName-count-zero";
        "كواشا ملاوي",
        "displayName-count-one";
        "كواشا ملاوي",
        "displayName-count-two";
        "كواشا ملاوي",
        "displayName-count-few";
        "كواشا ملاوي",
        "displayName-count-many";
        "كواشا ملاوي",
        "displayName-count-other";
        "كواشا ملاوي",
        "symbol";
        "MWK";
    }
    "MXN";
    {
        "displayName";
        "بيزو مكسيكي",

```

```

        "displayName-count-zero";
        "بيزو مكسيكي",
        "displayName-count-one";
        "بيزو مكسيكي",
        "displayName-count-two";
        "بيزو مكسيكي",
        "displayName-count-few";
        "بيزو مكسيكي",
        "displayName-count-many";
        "بيزو مكسيكي",
        "displayName-count-other";
        "بيزو مكسيكي",
        "symbol";
        "MX$";
        "symbol-alt-narrow";
        "MX$";
    }
    "MXP";
    {
        "displayName";
        "1992-1861 - بيزو فضي مكسيكي",
        "symbol";
        "MXP";
    }
    "MXV";
    {
        "displayName";
        "MXV",
        "symbol";
        "MXV";
    }
    "MYR";
    {
        "displayName";
        "رينغيت ماليزي",
        "displayName-count-zero";
        "رينغيت ماليزي",
        "displayName-count-one";
        "رينغيت ماليزي",
        "displayName-count-two";
        "رينغيت ماليزي",
        "displayName-count-few";
        "رينغيت ماليزي",
        "displayName-count-many";
        "رينغيت ماليزي",
        "displayName-count-other";
        "رينغيت ماليزي",
        "symbol";
        "MYR",
        "symbol-alt-narrow";
        "RM";
    }
    "MZE";
    {
        "displayName";
        "اسكود موزمبيق",
        "symbol";
    }

```



```

        "MZE";
    }
    "MZM";
    {
        "displayName";
        "MZM",
        "symbol";
        "MZM";
    }
    "MZN";
    {
        "displayName";
        "متكال موزمبيقي",
        "displayName-count-zero";
        "متكال موزمبيقي",
        "displayName-count-one";
        "متكال موزمبيقي",
        "displayName-count-two";
        "متكال موزمبيقي",
        "displayName-count-few";
        "متكال موزمبيقي",
        "displayName-count-many";
        "متكال موزمبيقي",
        "displayName-count-other";
        "متكال موزمبيقي",
        "symbol";
        "MZN";
    }
    "NAD";
    {
        "displayName";
        "دولار ناميبي",
        "displayName-count-zero";
        "دولار ناميبي",
        "displayName-count-one";
        "دولار ناميبي",
        "displayName-count-two";
        "دولار ناميبي",
        "displayName-count-few";
        "دولار ناميبي",
        "displayName-count-many";
        "دولار ناميبي",
        "displayName-count-other";
        "دولار ناميبي",
        "symbol";
        "NAD",
        "symbol-alt-narrow";
        "$";
    }
    "NGN";
    {
        "displayName";
        "نايرا نيجيري",
        "displayName-count-zero";
        "نايرا نيجيري",
        "displayName-count-one";
        "نايرا نيجيري",
    }

```

```

        "displayName-count-two";
        "نايرا نيجيري",
        "displayName-count-few";
        "نايرا نيجيري",
        "displayName-count-many";
        "نايرا نيجيري",
        "displayName-count-other";
        "نايرا نيجيري",
        "symbol";
        "NGN",
        "symbol-alt-narrow";
        "₦";
    }
    "NIC";
    {
        "displayName";
        "كوردوبه نيكاراغوا",
        "symbol";
        "NIC";
    }
    "NIO";
    {
        "displayName";
        "قرطبة نيكاراغوا",
        "displayName-count-zero";
        "قرطبة نيكاراغوا",
        "displayName-count-one";
        "قرطبة نيكاراغوا",
        "displayName-count-two";
        "قرطبة نيكاراغوا",
        "displayName-count-few";
        "قرطبة نيكاراغوا",
        "displayName-count-many";
        "قرطبة نيكاراغوا",
        "displayName-count-other";
        "قرطبة نيكاراغوا",
        "symbol";
        "NIO",
        "symbol-alt-narrow";
        "C$";
    }
    "NLG";
    {
        "displayName";
        "جلدر هولندي",
        "symbol";
        "NLG";
    }
    "NOK";
    {
        "displayName";
        "كرونة نرويجية",
        "displayName-count-zero";
        "كرونة نرويجية",
        "displayName-count-one";
        "كرونة نرويجية",
        "displayName-count-two";
    }

```

```

        "كرونة نرويجية",
        "displayName-count-few";
        "كرونة نرويجية",
        "displayName-count-many";
        "كرونة نرويجية",
        "displayName-count-other";
        "كرونة نرويجية",
        "symbol";
        "NOK",
        "symbol-alt-narrow";
        "kr";
    }
    "NPR";
    {
        "displayName";
        "روبية نيبالي",
        "displayName-count-zero";
        "روبية نيبالي",
        "displayName-count-one";
        "روبية نيبالي",
        "displayName-count-two";
        "روبية نيبالي",
        "displayName-count-few";
        "روبية نيبالي",
        "displayName-count-many";
        "روبية نيبالي",
        "displayName-count-other";
        "روبية نيبالي",
        "symbol";
        "NPR",
        "symbol-alt-narrow";
        "Rs";
    }
    "NZD";
    {
        "displayName";
        "دولار نيوزيلندي",
        "displayName-count-zero";
        "دولار نيوزيلندي",
        "displayName-count-one";
        "دولار نيوزيلندي",
        "displayName-count-two";
        "دولار نيوزيلندي",
        "displayName-count-few";
        "دولار نيوزيلندي",
        "displayName-count-many";
        "دولار نيوزيلندي",
        "displayName-count-other";
        "دولار نيوزيلندي",
        "symbol";
        "NZ$",
        "symbol-alt-narrow";
        "NZ$";
    }
    "OMR";
    {
        "displayName";

```

```

        "ول عماني",
        "displayName-count-zero";
        "ول عماني",
        "displayName-count-one";
        "ول عماني",
        "displayName-count-two";
        "ول عماني",
        "displayName-count-few";
        "ول عماني",
        "displayName-count-many";
        "ول عماني",
        "displayName-count-other";
        "ول عماني",
        "symbol";
        ".ر.ع.";
    }
    "PAB";
    {
        "displayName";
        "بالبوا بنمي",
        "displayName-count-zero";
        "بالبوا بنمي",
        "displayName-count-one";
        "بالبوا بنمي",
        "displayName-count-two";
        "بالبوا بنمي",
        "displayName-count-few";
        "بالبوا بنمي",
        "displayName-count-many";
        "بالبوا بنمي",
        "displayName-count-other";
        "بالبوا بنمي",
        "symbol";
        "PAB";
    }
    "PEI";
    {
        "displayName";
        "PEI",
        "symbol";
        "PEI";
    }
    "PEN";
    {
        "displayName";
        "سول جديد البيرو",
        "displayName-count-zero";
        "سول جديد البيرو",
        "displayName-count-one";
        "سول جديد البيرو",
        "displayName-count-two";
        "سول جديد البيرو",
        "displayName-count-few";
        "سول جديد البيرو",
        "displayName-count-many";
        "سول جديد البيرو",
        "displayName-count-other";
    }

```

```

        "سول جديد البيرو",
        "symbol";
        "PEN";
    }
    "PES";
    {
        "displayName";
        "PES",
        "symbol";
        "PES";
    }
    "PGK";
    {
        "displayName";
        "كينا بابوا غينيا الجديدة",
        "displayName-count-zero";
        "كينا بابوا غينيا الجديدة",
        "displayName-count-one";
        "كينا بابوا غينيا الجديدة",
        "displayName-count-two";
        "كينا بابوا غينيا الجديدة",
        "displayName-count-few";
        "كينا بابوا غينيا الجديدة",
        "displayName-count-many";
        "كينا بابوا غينيا الجديدة",
        "displayName-count-other";
        "كينا بابوا غينيا الجديدة",
        "symbol";
        "PGK";
    }
    "PHP";
    {
        "displayName";
        "بيزو فلبيني",
        "displayName-count-zero";
        "بيزو فلبيني",
        "displayName-count-one";
        "بيزو فلبيني",
        "displayName-count-two";
        "بيزو فلبيني",
        "displayName-count-few";
        "بيزو فلبيني",
        "displayName-count-many";
        "بيزو فلبيني",
        "displayName-count-other";
        "بيزو فلبيني",
        "symbol";
        "PHP",
        "symbol-alt-narrow";
        "₱";
    }
    "PKR";
    {
        "displayName";
        "روبية باكستاني",
        "displayName-count-zero";
        "روبية باكستاني",

```

```

        "displayName-count-one";
        "روبيۃ باڪستاني",
        "displayName-count-two";
        "روبيۃ باڪستاني",
        "displayName-count-few";
        "روبيۃ باڪستاني",
        "displayName-count-many";
        "روبيۃ باڪستاني",
        "displayName-count-other";
        "روبيۃ باڪستاني",
        "symbol";
        "PKR",
        "symbol-alt-narrow";
        "Rs";
    }
    "PLN";
    {
        "displayName";
        "زلوتي بولندي",
        "displayName-count-zero";
        "زلوتي بولندي",
        "displayName-count-one";
        "زلوتي بولندي",
        "displayName-count-two";
        "زلوتي بولندي",
        "displayName-count-few";
        "زلوتي بولندي",
        "displayName-count-many";
        "زلوتي بولندي",
        "displayName-count-other";
        "زلوتي بولندي",
        "symbol";
        "PLN",
        "symbol-alt-narrow";
        "zł";
    }
    "PLZ";
    {
        "displayName";
        "زلوتي بولندي - 1950-1995",
        "symbol";
        "PLZ";
    }
    "PTE";
    {
        "displayName";
        "اسڪود پرتگالي",
        "symbol";
        "PTE";
    }
    "PYG";
    {
        "displayName";
        "جواراني پاراگواي",
        "displayName-count-zero";
        "جواراني پاراگواي",
        "displayName-count-one";
    }

```

```

        "جواراني باراجوای",
        "displayName-count-two";
        "جواراني باراجوای",
        "displayName-count-few";
        "جواراني باراجوای",
        "displayName-count-many";
        "جواراني باراجوای",
        "displayName-count-other";
        "جواراني باراجوای",
        "symbol";
        "PYG",
        "symbol-alt-narrow";
        "₲";
    }
    "QAR";
    {
        "displayName";
        "ڤل قطري",
        "displayName-count-zero";
        "ڤل قطري",
        "displayName-count-one";
        "ڤل قطري",
        "displayName-count-two";
        "ڤل قطري",
        "displayName-count-few";
        "ڤل قطري",
        "displayName-count-many";
        "ڤل قطري",
        "displayName-count-other";
        "ڤل قطري",
        "symbol";
        "ر.ق.";
    }
    "RHD";
    {
        "displayName";
        "ڊولار روڊيسي",
        "symbol";
        "RHD";
    }
    "ROL";
    {
        "displayName";
        "ليو روماني قديم",
        "symbol";
        "ROL";
    }
    "RON";
    {
        "displayName";
        "ليو روماني",
        "displayName-count-zero";
        "ليو روماني",
        "displayName-count-one";
        "ليو روماني",
        "displayName-count-two";
        "ليو روماني",
    }

```

```

        "displayName-count-few";
        "ليو روماني",
        "displayName-count-many";
        "ليو روماني",
        "displayName-count-other";
        "ليو روماني",
        "symbol";
        "RON",
        "symbol-alt-narrow";
        "lei";
    }
    "RSD";
    {
        "displayName";
        "دينار صربي",
        "displayName-count-zero";
        "دينار صربي",
        "displayName-count-one";
        "دينار صربي",
        "displayName-count-two";
        "ديناران صربيان",
        "displayName-count-few";
        "دينارات صربية",
        "displayName-count-many";
        "دينارًا صربيًا",
        "displayName-count-other";
        "دينار صربي",
        "symbol";
        "RSD";
    }
    "RUB";
    {
        "displayName";
        "روبل روسي",
        "displayName-count-zero";
        "روبل روسي",
        "displayName-count-one";
        "روبل روسي",
        "displayName-count-two";
        "روبل روسي",
        "displayName-count-few";
        "روبل روسي",
        "displayName-count-many";
        "روبل روسي",
        "displayName-count-other";
        "روبل روسي",
        "symbol";
        "RUB",
        "symbol-alt-narrow";
        "₽";
    }
    "RUR";
    {
        "displayName";
        "1998-1991 - روبل روسي",
        "symbol";
        "RUR",

```



```

        "symbol-alt-narrow";
        "p.";
    }
    "RWF";
    {
        "displayName";
        "فرنك رواندي",
        "displayName-count-zero";
        "فرنك رواندي",
        "displayName-count-one";
        "فرنك رواندي",
        "displayName-count-two";
        "فرنك رواندي",
        "displayName-count-few";
        "فرنك رواندي",
        "displayName-count-many";
        "فرنك رواندي",
        "displayName-count-other";
        "فرنك رواندي",
        "symbol";
        "RWF",
        "symbol-alt-narrow";
        "RF";
    }
    "SAR";
    {
        "displayName";
        "رول سعودي",
        "displayName-count-zero";
        "رول سعودي",
        "displayName-count-one";
        "رول سعودي",
        "displayName-count-two";
        "رول سعودي",
        "displayName-count-few";
        "رول سعودي",
        "displayName-count-many";
        "رول سعودي",
        "displayName-count-other";
        "رول سعودي",
        "symbol";
        "ر.س.";
    }
    "SBD";
    {
        "displayName";
        "دولار جزر سليمان",
        "displayName-count-zero";
        "دولار جزر سليمان",
        "displayName-count-one";
        "دولار جزر سليمان",
        "displayName-count-two";
        "دولار جزر سليمان",
        "displayName-count-few";
        "دولار جزر سليمان",
        "displayName-count-many";
        "دولار جزر سليمان",
    }

```

```

        "displayName-count-other";
        "دولار جزر سليمان",
        "symbol";
        "SBD",
        "symbol-alt-narrow";
        "SB$";
    }
    "SCR";
    {
        "displayName";
        "روبية سيشيلية",
        "displayName-count-zero";
        "روبية سيشيلية",
        "displayName-count-one";
        "روبية سيشيلية",
        "displayName-count-two";
        "روبية سيشيلية",
        "displayName-count-few";
        "روبية سيشيلية",
        "displayName-count-many";
        "روبية سيشيلية",
        "displayName-count-other";
        "روبية سيشيلية",
        "symbol";
        "SCR";
    }
    "SDD";
    {
        "displayName";
        "دينار سوداني",
        "symbol";
        "د.س.";
    }
    "SDG";
    {
        "displayName";
        "جنيه سوداني",
        "displayName-count-zero";
        "جنيه سوداني",
        "displayName-count-one";
        "جنيه سوداني",
        "displayName-count-two";
        "جنيه سوداني",
        "displayName-count-few";
        "جنيهات سودانية",
        "displayName-count-many";
        "جنيهاً سودانياً",
        "displayName-count-other";
        "جنيه سوداني",
        "symbol";
        "ج.س.";
    }
    "SDP";
    {
        "displayName";
        "جنيه سوداني قديم",
        "symbol";
    }

```

```

        "SDP";
    }
    "SEK";
    {
        "displayName";
        "كرونة سويدية",
        "displayName-count-zero";
        "كرونة سويدية",
        "displayName-count-one";
        "كرونة سويدية",
        "displayName-count-two";
        "كرونة سويدية",
        "displayName-count-few";
        "كرونة سويدية",
        "displayName-count-many";
        "كرونة سويدية",
        "displayName-count-other";
        "كرونة سويدية",
        "symbol";
        "SEK",
        "symbol-alt-narrow";
        "kr";
    }
    "SGD";
    {
        "displayName";
        "دولار سنغافوري",
        "displayName-count-zero";
        "دولار سنغافوري",
        "displayName-count-one";
        "دولار سنغافوري",
        "displayName-count-two";
        "دولار سنغافوري",
        "displayName-count-few";
        "دولار سنغافوري",
        "displayName-count-many";
        "دولار سنغافوري",
        "displayName-count-other";
        "دولار سنغافوري",
        "symbol";
        "SGD",
        "symbol-alt-narrow";
        "$";
    }
    "SHP";
    {
        "displayName";
        "جنيه سانت هيلين",
        "displayName-count-zero";
        "جنيه سانت هيلين",
        "displayName-count-one";
        "جنيه سانت هيلين",
        "displayName-count-two";
        "جنيه سانت هيلين",
        "displayName-count-few";
        "جنيه سانت هيلين",
        "displayName-count-many";
    }

```

```

        "جنيه سانت هيلين",
        "displayName-count-other";
        "جنيه سانت هيلين",
        "symbol";
        "SHP",
        "symbol-alt-narrow";
        "£";
    }
    "SIT";
    {
        "displayName";
        "تولار سلوفيني",
        "symbol";
        "SIT";
    }
    "SKK";
    {
        "displayName";
        "كرونه سلوفاكية",
        "symbol";
        "SKK";
    }
    "SLL";
    {
        "displayName";
        "ليون سيراليوني",
        "displayName-count-zero";
        "ليون سيراليوني",
        "displayName-count-one";
        "ليون سيراليوني",
        "displayName-count-two";
        "ليون سيراليوني",
        "displayName-count-few";
        "ليون سيراليوني",
        "displayName-count-many";
        "ليون سيراليوني",
        "displayName-count-other";
        "ليون سيراليوني",
        "symbol";
        "SLL";
    }
    "SOS";
    {
        "displayName";
        "شلن صومالي",
        "displayName-count-zero";
        "شلن صومالي",
        "displayName-count-one";
        "شلن صومالي",
        "displayName-count-two";
        "شلن صومالي",
        "displayName-count-few";
        "شلن صومالي",
        "displayName-count-many";
        "شلن صومالي",
        "displayName-count-other";
        "شلن صومالي",
    }

```

```

        "symbol";
        "SOS";
    }
    "SRD";
    {
        "displayName";
        "دولار سورينامي",
        "displayName-count-zero";
        "دولار سورينامي",
        "displayName-count-one";
        "دولار سورينامي",
        "displayName-count-two";
        "دولار سورينامي",
        "displayName-count-few";
        "دولار سورينامي",
        "displayName-count-many";
        "دولار سورينامي",
        "displayName-count-other";
        "دولار سورينامي",
        "symbol";
        "SRD",
        "symbol-alt-narrow";
        "SR$";
    }
    "SRG";
    {
        "displayName";
        "جلدر سورينامي",
        "symbol";
        "SRG";
    }
    "SSP";
    {
        "displayName";
        "جنيه جنوب السودان",
        "displayName-count-zero";
        "جنيه جنوب السودان",
        "displayName-count-one";
        "جنيه جنوب السودان",
        "displayName-count-two";
        "جنيهان جنوب السودان",
        "displayName-count-few";
        "جنيهات جنوب السودان",
        "displayName-count-many";
        "جنيها جنوب السودان",
        "displayName-count-other";
        "جنيه جنوب السودان",
        "symbol";
        "SSP",
        "symbol-alt-narrow";
        "£";
    }
    "STD";
    {
        "displayName";
        "دوبرا ساو تومي وبرينسيبي",
        "displayName-count-zero";

```

```

        "دوبرا ساو تومي وبرينسيبي",
        "displayName-count-one";
        "دوبرا ساو تومي وبرينسيبي",
        "displayName-count-two";
        "دوبرا ساو تومي وبرينسيبي",
        "displayName-count-few";
        "دوبرا ساو تومي وبرينسيبي",
        "displayName-count-many";
        "دوبرا ساو تومي وبرينسيبي",
        "displayName-count-other";
        "دوبرا ساو تومي وبرينسيبي",
        "symbol";
        "STD",
        "symbol-alt-narrow";
        "Db";
    }
    "SUR";
    {
        "displayName";
        "روبل سوفيتي",
        "symbol";
        "SUR";
    }
    "SVC";
    {
        "displayName";
        "كولون سلغادوري",
        "symbol";
        "SVC";
    }
    "SYP";
    {
        "displayName";
        "ليرة سورية",
        "displayName-count-zero";
        "ليرة سورية",
        "displayName-count-one";
        "ليرة سورية",
        "displayName-count-two";
        "ليرة سورية",
        "displayName-count-few";
        "ليرة سورية",
        "displayName-count-many";
        "ليرة سورية",
        "displayName-count-other";
        "ليرة سورية",
        "symbol";
        "ل.س.",
        "symbol-alt-narrow";
        "£";
    }
    "SZL";
    {
        "displayName";
        "ليلانجيني سوازيلندي",
        "displayName-count-zero";
        "ليلانجيني سوازيلندي",

```

```

        "displayName-count-one";
        "لِيلَانْجِينِي سَوَازِيلَنْدِي",
        "displayName-count-two";
        "لِيلَانْجِينِي سَوَازِيلَنْدِي",
        "displayName-count-few";
        "لِيلَانْجِينِي سَوَازِيلَنْدِي",
        "displayName-count-many";
        "لِيلَانْجِينِي سَوَازِيلَنْدِي",
        "displayName-count-other";
        "لِيلَانْجِينِي سَوَازِيلَنْدِي",
        "symbol";
        "SZL";
    }
    "THB";
    {
        "displayName";
        "بَاخْت تَايْلَانْدِي",
        "displayName-count-zero";
        "بَاخْت تَايْلَانْدِي",
        "displayName-count-one";
        "بَاخْت تَايْلَانْدِي",
        "displayName-count-two";
        "بَاخْت تَايْلَانْدِي",
        "displayName-count-few";
        "بَاخْت تَايْلَانْدِي",
        "displayName-count-many";
        "بَاخْت تَايْلَانْدِي",
        "displayName-count-other";
        "بَاخْت تَايْلَانْدِي",
        "symbol";
        "฿",
        "symbol-alt-narrow";
        "฿";
    }
    "TJR";
    {
        "displayName";
        "رُوبَل طَاچِيكْسْتَانِي",
        "symbol";
        "TJR";
    }
    "TJS";
    {
        "displayName";
        "سُومُونِي طَاچِيكْسْتَانِي",
        "displayName-count-zero";
        "سُومُونِي طَاچِيكْسْتَانِي",
        "displayName-count-one";
        "سُومُونِي طَاچِيكْسْتَانِي",
        "displayName-count-two";
        "سُومُونِي طَاچِيكْسْتَانِي",
        "displayName-count-few";
        "سُومُونِي طَاچِيكْسْتَانِي",
        "displayName-count-many";
        "سُومُونِي طَاچِيكْسْتَانِي",
        "displayName-count-other";
        "سُومُونِي طَاچِيكْسْتَانِي",
    }

```

```

        "symbol";
        "TJS";
    }
    "TMM";
    {
        "displayName";
        "مانات ترکمنستاني",
        "symbol";
        "TMM";
    }
    "TMT";
    {
        "displayName";
        "مانات ترکمانستان",
        "displayName-count-zero";
        "مانات ترکمانستان",
        "displayName-count-one";
        "مانات ترکمانستان",
        "displayName-count-two";
        "مانات ترکمانستان",
        "displayName-count-few";
        "مانات ترکمانستان",
        "displayName-count-many";
        "مانات ترکمانستان",
        "displayName-count-other";
        "مانات ترکمانستان",
        "symbol";
        "TMT";
    }
    "TND";
    {
        "displayName";
        "دينار تونسي",
        "displayName-count-zero";
        "دينار تونسي",
        "displayName-count-one";
        "دينار تونسي",
        "displayName-count-two";
        "ديناران تونسيان",
        "displayName-count-few";
        "دينارات تونسية",
        "displayName-count-many";
        "دينارًا تونسيًا",
        "displayName-count-other";
        "دينار تونسي",
        "symbol";
        "د.ت.";
    }
    "TOP";
    {
        "displayName";
        "بانغا تونغا",
        "displayName-count-zero";
        "بانغا تونغا",
        "displayName-count-one";
        "بانغا تونغا",
        "displayName-count-two";
    }

```



```

        "بانغا تونغا",
        "displayName-count-few";
        "بانغا تونغا",
        "displayName-count-many";
        "بانغا تونغا",
        "displayName-count-other";
        "بانغا تونغا",
        "symbol";
        "TOP",
        "symbol-alt-narrow";
        "T$";
    }
    "TPE";
    {
        "displayName";
        "اسكود تيموري",
        "symbol";
        "TPE";
    }
    "TRL";
    {
        "displayName";
        "ليرة تركي",
        "symbol";
        "TRL";
    }
    "TRY";
    {
        "displayName";
        "ليرة تركية",
        "displayName-count-zero";
        "ليرة تركية",
        "displayName-count-one";
        "ليرة تركية",
        "displayName-count-two";
        "ليرة تركية",
        "displayName-count-few";
        "ليرة تركية",
        "displayName-count-many";
        "ليرة تركية",
        "displayName-count-other";
        "ليرة تركية",
        "symbol";
        "TRY",
        "symbol-alt-narrow";
        "₺",
        "symbol-alt-variant";
        "TL";
    }
    "TTD";
    {
        "displayName";
        "دولار ترينداد وتوباغو",
        "displayName-count-zero";
        "دولار ترينداد وتوباغو",
        "displayName-count-one";
        "دولار ترينداد وتوباغو",

```

```

        "displayName-count-two";
        "دولار ترينداد وتوباجو",
        "displayName-count-few";
        "دولار ترينداد وتوباجو",
        "displayName-count-many";
        "دولار ترينداد وتوباجو",
        "displayName-count-other";
        "دولار ترينداد وتوباجو",
        "symbol";
        "TTD",
        "symbol-alt-narrow";
        "TT$";
    }
    "TWD";
    {
        "displayName";
        "دولار تايواني",
        "displayName-count-zero";
        "دولار تايواني",
        "displayName-count-one";
        "دولار تايواني",
        "displayName-count-two";
        "دولار تايواني",
        "displayName-count-few";
        "دولار تايواني",
        "displayName-count-many";
        "دولار تايواني",
        "displayName-count-other";
        "دولار تايواني",
        "symbol";
        "NT$";
        "symbol-alt-narrow";
        "NT$";
    }
    "TZS";
    {
        "displayName";
        "شلن تنزاني",
        "displayName-count-zero";
        "شلن تنزاني",
        "displayName-count-one";
        "شلن تنزاني",
        "displayName-count-two";
        "شلن تنزاني",
        "displayName-count-few";
        "شلن تنزاني",
        "displayName-count-many";
        "شلن تنزاني",
        "displayName-count-other";
        "شلن تنزاني",
        "symbol";
        "TZS";
    }
    "UAH";
    {
        "displayName";
        "هريفنيا اوكراني",

```

```

        "displayName-count-zero";
        "هريفنيا أوكراڻي",
        "displayName-count-one";
        "هريفنيا أوكراڻي",
        "displayName-count-two";
        "هريفنيا أوكراڻي",
        "displayName-count-few";
        "هريفنيا أوكراڻي",
        "displayName-count-many";
        "هريفنيا أوكراڻي",
        "displayName-count-other";
        "هريفنيا أوكراڻي",
        "symbol";
        "UAH",
        "symbol-alt-narrow";
        "₴";
    }
    "UAK";
    {
        "displayName";
        "UAK",
        "symbol";
        "UAK";
    }
    "UGS";
    {
        "displayName";
        "شلن أوغندي - 1966-1987",
        "symbol";
        "UGS";
    }
    "UGX";
    {
        "displayName";
        "شلن أوغندي",
        "displayName-count-zero";
        "شلن أوغندي",
        "displayName-count-one";
        "شلن أوغندي",
        "displayName-count-two";
        "شلن أوغندي",
        "displayName-count-few";
        "شلن أوغندي",
        "displayName-count-many";
        "شلن أوغندي",
        "displayName-count-other";
        "شلن أوغندي",
        "symbol";
        "UGX";
    }
    "USD";
    {
        "displayName";
        "دولار أمريكي",
        "displayName-count-zero";
        "دولار أمريكي",
        "displayName-count-one";
    }

```

```

        "دولار أمريكي",
        "displayName-count-two";
        "دولار أمريكي",
        "displayName-count-few";
        "دولار أمريكي",
        "displayName-count-many";
        "دولار أمريكي",
        "displayName-count-other";
        "دولار أمريكي",
        "symbol";
        "US$";
        "symbol-alt-narrow";
        "US$";
    }
    "USN";
    {
        "displayName";
        "دولار أمريكي (اليوم التالي)",
        "symbol";
        "USN";
    }
    "USS";
    {
        "displayName";
        "دولار أمريكي (نفس اليوم)",
        "symbol";
        "USS";
    }
    "UYI";
    {
        "displayName";
        "UYI",
        "symbol";
        "UYI";
    }
    "UYP";
    {
        "displayName";
        "بيزو أوروغواي - 1993-1975",
        "symbol";
        "UYP";
    }
    "UYU";
    {
        "displayName";
        "بيزو أوروغواي",
        "displayName-count-zero";
        "بيزو أوروغواي",
        "displayName-count-one";
        "بيزو أوروغواي",
        "displayName-count-two";
        "بيزو أوروغواي",
        "displayName-count-few";
        "بيزو أوروغواي",
        "displayName-count-many";
        "بيزو أوروغواي",
        "displayName-count-other";
    }

```

```

        "بیزو اوروغوای",
        "symbol";
    "UYU",
        "symbol-alt-narrow";
    "UY$";
}
"UZS";
{
    "displayName";
    "سوم اۆزبکستانی",
        "displayName-count-zero";
    "سوم اۆزبکستانی",
        "displayName-count-one";
    "سوم اۆزبکستانی",
        "displayName-count-two";
    "سوم اۆزبکستانی",
        "displayName-count-few";
    "سوم اۆزبکستانی",
        "displayName-count-many";
    "سوم اۆزبکستانی",
        "displayName-count-other";
    "سوم اۆزبکستانی",
        "symbol";
    "UZS";
}
"VEB";
{
    "displayName";
    "بولیفار فنزویلی - 2008-1871",
        "symbol";
    "VEB";
}
"VEF";
{
    "displayName";
    "بولیفار فنزویلی",
        "displayName-count-zero";
    "بولیفار فنزویلی",
        "displayName-count-one";
    "بولیفار فنزویلی",
        "displayName-count-two";
    "بولیفار فنزویلی",
        "displayName-count-few";
    "بولیفار فنزویلی",
        "displayName-count-many";
    "بولیفار فنزویلی",
        "displayName-count-other";
    "بولیفار فنزویلی",
        "symbol";
    "VEF",
        "symbol-alt-narrow";
    "Bs";
}
"VND";
{
    "displayName";
    "دونچ فیتنامی",

```

```

        "displayName-count-zero";
        "دونج فيتنامي",
        "displayName-count-one";
        "دونج فيتنامي",
        "displayName-count-two";
        "دونج فيتنامي",
        "displayName-count-few";
        "دونج فيتنامي",
        "displayName-count-many";
        "دونج فيتنامي",
        "displayName-count-other";
        "دونج فيتنامي",
        "symbol";
        "₫",
        "symbol-alt-narrow";
        "₫";
    }
    "VNN";
    {
        "displayName";
        "VNN",
        "symbol";
        "VNN";
    }
    "VUV";
    {
        "displayName";
        "فاتو فانواتو",
        "displayName-count-zero";
        "فاتو فانواتو",
        "displayName-count-one";
        "فاتو فانواتو",
        "displayName-count-two";
        "فاتو فانواتو",
        "displayName-count-few";
        "فاتو فانواتو",
        "displayName-count-many";
        "فاتو فانواتو",
        "displayName-count-other";
        "فاتو فانواتو",
        "symbol";
        "VUV";
    }
    "WST";
    {
        "displayName";
        "تالا ساموا",
        "displayName-count-zero";
        "تالا ساموا",
        "displayName-count-one";
        "تالا ساموا",
        "displayName-count-two";
        "تالا ساموا",
        "displayName-count-few";
        "تالا ساموا",
        "displayName-count-many";
        "تالا ساموا",
    }

```

```

        "displayName-count-other";
        "تالا ساموا",
        "symbol";
        "WST";
    }
    "XAF";
    {
        "displayName";
        "فرنك وسط أفريقي",
        "displayName-count-zero";
        "فرنك وسط أفريقي",
        "displayName-count-one";
        "فرنك وسط أفريقي",
        "displayName-count-two";
        "فرنك وسط أفريقي",
        "displayName-count-few";
        "فرنك وسط أفريقي",
        "displayName-count-many";
        "فرنك وسط أفريقي",
        "displayName-count-other";
        "فرنك وسط أفريقي",
        "symbol";
        "FCFA";
    }
    "XAG";
    {
        "displayName";
        "فضة",
        "symbol";
        "XAG";
    }
    "XAU";
    {
        "displayName";
        "ذهب",
        "symbol";
        "XAU";
    }
    "XBA";
    {
        "displayName";
        "الوحدة الأوروبية المركبة",
        "symbol";
        "XBA";
    }
    "XBB";
    {
        "displayName";
        "الوحدة المالية الأوروبية",
        "symbol";
        "XBB";
    }
    "XBC";
    {
        "displayName";
        "الوحدة الحسابية الأوروبية",
        "symbol";
    }

```

```

        "XBC";
    }
    "XBD";
    {
        "displayName";
        "وحدة الحساب الأوروبية (XBD)",
        "symbol";
        "XBD";
    }
    "XCD";
    {
        "displayName";
        "دولار شرق الكاريبي",
        "displayName-count-zero";
        "دولار شرق الكاريبي",
        "displayName-count-one";
        "دولار شرق الكاريبي",
        "displayName-count-two";
        "دولار شرق الكاريبي",
        "displayName-count-few";
        "دولار شرق الكاريبي",
        "displayName-count-many";
        "دولار شرق الكاريبي",
        "displayName-count-other";
        "دولار شرق الكاريبي",
        "symbol";
        "EC$";
        "symbol-alt-narrow";
        "$";
    }
    "XDR";
    {
        "displayName";
        "حقوق السحب الخاصة",
        "symbol";
        "XDR";
    }
    "XEU";
    {
        "displayName";
        "وحدة النقد الأوروبية",
        "symbol";
        "XEU";
    }
    "XFO";
    {
        "displayName";
        "فرنك فرنسي ذهبي",
        "symbol";
        "XFO";
    }
    "XFU";
    {
        "displayName";
        "فرنك فرنسي (UIC)",
        "symbol";
        "XFU";
    }

```



```

    }
    "XOF";
    {
        "displayName";
        "فرنك غرب أفريقي",
        "displayName-count-zero";
        "فرنك غرب أفريقي",
        "displayName-count-one";
        "فرنك غرب أفريقي",
        "displayName-count-two";
        "فرنك غرب أفريقي",
        "displayName-count-few";
        "فرنك غرب أفريقي",
        "displayName-count-many";
        "فرنك غرب أفريقي",
        "displayName-count-other";
        "فرنك غرب أفريقي",
        "symbol";
        "CFA";
    }
    "XPD";
    {
        "displayName";
        "بالاديوم",
        "symbol";
        "XPD";
    }
    "XPF";
    {
        "displayName";
        "فرنك سي إف بي",
        "displayName-count-zero";
        "فرنك سي إف بي",
        "displayName-count-one";
        "فرنك سي إف بي",
        "displayName-count-two";
        "فرنك سي إف بي",
        "displayName-count-few";
        "فرنك سي إف بي",
        "displayName-count-many";
        "فرنك سي إف بي",
        "displayName-count-other";
        "فرنك سي إف بي",
        "symbol";
        "CFPF";
    }
    "XPT";
    {
        "displayName";
        "البلاطين",
        "symbol";
        "XPT";
    }
    "XRE";
    {
        "displayName";
        "XRE",

```

```

        "symbol";
        "XRE";
    }
    "XSU";
    {
        "displayName";
        "XSU",
        "symbol";
        "XSU";
    }
    "XTS";
    {
        "displayName";
        "كود اختبار العملة",
        "symbol";
        "XTS";
    }
    "XUA";
    {
        "displayName";
        "XUA",
        "symbol";
        "XUA";
    }
    "XXX";
    {
        "displayName";
        "عملة غير معروفة",
        "displayName-count-zero";
        "(عملة غير معروفة)",
        "displayName-count-one";
        "(عملة غير معروفة)",
        "displayName-count-two";
        "(عملة غير معروفة)",
        "displayName-count-few";
        "(عملة غير معروفة)",
        "displayName-count-many";
        "(عملة غير معروفة)",
        "displayName-count-other";
        "(عملة غير معروفة)",
        "symbol";
        "***";
    }
    "YDD";
    {
        "displayName";
        "دينار يمني",
        "symbol";
        "YDD";
    }
    "YER";
    {
        "displayName";
        "رل يمني",
        "displayName-count-zero";
        "رل يمني",
        "displayName-count-one";
    }

```

```

        "ول يمني",
        "displayName-count-two";
        "ول يمني",
        "displayName-count-few";
        "ول يمني",
        "displayName-count-many";
        "ول يمني",
        "displayName-count-other";
        "ول يمني",
        "symbol";
        "ر.ي.";
    }
    "YUD";
    {
        "displayName";
        "دينار يوغسلافي",
        "symbol";
        "YUD";
    }
    "YUM";
    {
        "displayName";
        "YUM",
        "symbol";
        "YUM";
    }
    "YUN";
    {
        "displayName";
        "دينار يوغسلافي قابل للتحويل",
        "symbol";
        "YUN";
    }
    "YUR";
    {
        "displayName";
        "YUR",
        "symbol";
        "YUR";
    }
    "ZAL";
    {
        "displayName";
        "راند جنوب أفريقيا -مالي",
        "symbol";
        "ZAL";
    }
    "ZAR";
    {
        "displayName";
        "راند جنوب أفريقيا",
        "displayName-count-zero";
        "راند جنوب أفريقيا",
        "displayName-count-one";
        "راند جنوب أفريقيا",
        "displayName-count-two";
        "راند جنوب أفريقيا",

```

```

        "displayName-count-few";
        "راند جنوب أفريقيا",
        "displayName-count-many";
        "راند جنوب أفريقيا",
        "displayName-count-other";
        "راند جنوب أفريقيا",
        "symbol";
        "ZAR",
        "symbol-alt-narrow";
        "R";
    }
    "ZMK";
    {
        "displayName";
        "2012-1968 - كواشا زامبي",
        "displayName-count-zero";
        "2012-1968 - كواشا زامبي",
        "displayName-count-one";
        "2012-1968 - كواشا زامبي",
        "displayName-count-two";
        "2012-1968 - كواشا زامبي",
        "displayName-count-few";
        "2012-1968 - كواشا زامبي",
        "displayName-count-many";
        "2012-1968 - كواشا زامبي",
        "displayName-count-other";
        "2012-1968 - كواشا زامبي",
        "symbol";
        "ZMK";
    }
    "ZMW";
    {
        "displayName";
        "كواشا زامبي",
        "displayName-count-zero";
        "كواشا زامبي",
        "displayName-count-one";
        "كواشا زامبي",
        "displayName-count-two";
        "كواشا زامبي",
        "displayName-count-few";
        "كواشا زامبي",
        "displayName-count-many";
        "كواشا زامبي",
        "displayName-count-other";
        "كواشا زامبي",
        "symbol";
        "ZMW",
        "symbol-alt-narrow";
        "ZK";
    }
    "ZRN";
    {
        "displayName";
        "زائير زائيري جديد",
        "symbol";
        "ZRN";
    }

```

```

    }
    "ZRW";
    {
        "displayName";
        "زائير زائيري",
        "symbol";
        "ZRW";
    }
    "ZWD";
    {
        "displayName";
        "دولار زمبابوي",
        "symbol";
        "ZWD";
    }
    "ZWL";
    {
        "displayName";
        "دولار زمبابوي 2009",
        "symbol";
        "ZWL";
    }
    "ZWR";
    {
        "displayName";
        "ZWR",
        "symbol";
        "ZWR";
    }
}

}

}

}
```

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
//import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import { loadCldr, L10n } from '@syncfusion/ej2-base';
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as timeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames);
L10n.load({
    'ar': {
        'calendar': {
            today: "اليوم"
        }
    }
});
class App extends React.Component {
```

```

    render() {
        return <CalendarComponent id="calendar" locale='ar'
enableRtl={true}/>;
    }
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
//import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import { loadCldr, L10n } from '@syncfusion/ej2-base';
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as timeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames);
L10n.load({
    'ar': {
        'calendar': {
            today: "اليوم"
        }
    }
});
class App extends React.Component<{}, {}> {
    render() {
        return <CalendarComponent id="calendar" locale='ar' enableRtl={true}
/>
    }
};
ReactDOM.render(<App />, document.getElementById('element'));

```

NUMBERINGSYSTEMS.JSON

```

{
  "supplemental": {
    "version": {
      "_number": "$Revision: 12732 $",
      "_unicodeVersion": "9.0.0",
      "_cldrVersion": "31"
    },
    "numberingSystems": {
      "adlm": {
        "_digits": "ᲀᲁᲂᲃᲄᲅᲆᲇᲈᲉᲐᲑᲒᲓᲔᲕᲖᲗᲘᲙᲚᲛᲜᲝᲞᲟᲠᲡᲢᲣᲤᲥᲦᲧᲨᲩᲪᲫᲬᲭᲮᲯᲰᲱᲲᲳᲴᲵᲶᲷᲸᲹᲺ᲻᲼ᲽᲾᲿ",
        "_type": "numeric"
      },
      "ahom": {
        "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱰᱱᱲᱳᱴᱵᱶᱷᱸᱹᱺᱻᱼᱽ᱾᱿",
        "_type": "numeric"
      },
      "arab": {

```

```

    "_digits": "٠١٢٣٤٥٦٧٨٩",
    "_type": "numeric"
  },
  "arabext": {
    "_digits": "٠١٢٣٤٥٦٧٨٩",
    "_type": "numeric"
  },
  "armn": {
    "_rules": "armenian-upper",
    "_type": "algorithmic"
  },
  "armnlow": {
    "_rules": "armenian-lower",
    "_type": "algorithmic"
  },
  "bali": {
    "_digits": "ꦏꦴꦩꦸꦥꦸꦏꦸꦩꦸꦥꦸꦏꦸꦩꦸꦥꦸꦏꦸ",
    "_type": "numeric"
  },
  "beng": {
    "_digits": "০১২৩৪৫৬৭৮৯",
    "_type": "numeric"
  },
  "bhks": {
    "_digits": "ꠌꠕꠗꠘꠙꠚꠛꠜꠝꠞꠟ",
    "_type": "numeric"
  },
  "brah": {
    "_digits": "୦୧୨୩୪୫୬୭୮୯",
    "_type": "numeric"
  },
  "cakm": {
    "_digits": "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
    "_type": "numeric"
  },
  "cham": {
    "_digits": "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
    "_type": "numeric"
  },
  "cyrl": {
    "_rules": "cyrillic-lower",
    "_type": "algorithmic"
  },
  "deva": {
    "_digits": "०१२३४५६७८९",
    "_type": "numeric"
  },
  "ethi": {
    "_rules": "ethiopic",
    "_type": "algorithmic"
  },
  "fullwide": {
    "_digits": "0 1 2 3 4 5 6 7 8 9",
    "_type": "numeric"
  },
  "geor": {

```

```

    "_rules": "georgian",
    "_type": "algorithmic"
  },
  "grek": {
    "_rules": "greek-upper",
    "_type": "algorithmic"
  },
  "greklow": {
    "_rules": "greek-lower",
    "_type": "algorithmic"
  },
  "gujr": {
    "_digits": "૦૧૨૩૪૫૬૭૮૯",
    "_type": "numeric"
  },
  "guru": {
    "_digits": "੦੧੨੩੪੫੬੭੮੯",
    "_type": "numeric"
  },
  "hanidays": {
    "_rules": "zh/SpelloutRules/spellout-numbering-days",
    "_type": "algorithmic"
  },
  "hanidec": {
    "_digits": "〇一二三四五六七八九",
    "_type": "numeric"
  },
  "hans": {
    "_rules": "zh/SpelloutRules/spellout-cardinal",
    "_type": "algorithmic"
  },
  "hansfin": {
    "_rules": "zh/SpelloutRules/spellout-cardinal-financial",
    "_type": "algorithmic"
  },
  "hant": {
    "_rules": "zh_Hant/SpelloutRules/spellout-cardinal",
    "_type": "algorithmic"
  },
  "hantfin": {
    "_rules": "zh_Hant/SpelloutRules/spellout-cardinal-financial",
    "_type": "algorithmic"
  },
  "hebr": {
    "_rules": "hebrew",
    "_type": "algorithmic"
  },
  "hmng": {
    "_digits": "᠐᠑᠒᠓᠐ᠠᠨᠣᠭᠠᠨ",
    "_type": "numeric"
  },
  "java": {
    "_digits": "᠐ᠠᠨᠠᠭᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨ",
    "_type": "numeric"
  },

```



```

"jpan": {
  "_rules": "ja/SpelloutRules/spellout-cardinal",
  "_type": "algorithmic"
},
"jpanfin": {
  "_rules": "ja/SpelloutRules/spellout-cardinal-financial",
  "_type": "algorithmic"
},
"kali": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"khmr": {
  "_digits": "០១២៣៤៥៦៧៨៩",
  "_type": "numeric"
},
"knda": {
  "_digits": "೦೧೨೩೪೫೬೭೮೯",
  "_type": "numeric"
},
"lana": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"lanatham": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"laoo": {
  "_digits": "໐໑໒໓໔໕໖໗໘໑",
  "_type": "numeric"
},
"latn": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"lepc": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"limb": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"mathbold": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathdbl": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathmono": {
  "_digits": "0123456789",
  "_type": "numeric"
},

```

```

"mathsanb": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathsans": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mlym": {
  "_digits": "ഐഎംഎൽഒന്നുവുൻ",
  "_type": "numeric"
},
"modi": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"mong": {
  "_digits": "ᠮᠣᠩᠭᠡᠨᠠᠨᠢ",
  "_type": "numeric"
},
"mroo": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"mtei": {
  "_digits": "ႤႬႬႬႬႬႬႬႬႬႬႬ",
  "_type": "numeric"
},
"mymr": {
  "_digits": "၀၁၂၃၄၅၆၇၈၉",
  "_type": "numeric"
},
"mymrshan": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mymrtlng": {
  "_digits": "ႤႬႬႬႬႬႬႬႬႬႬႬ",
  "_type": "numeric"
},
"newa": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"nkoo": {
  "_digits": "ႤႬႬႬႬႬႬႬႬႬႬႬ",
  "_type": "numeric"
},
"olck": {
  "_digits": "ႤႬႬႬႬႬႬႬႬႬႬႬ",
  "_type": "numeric"
},
"orya": {

```

```

    "_digits": "௦௧௨௩௪௫௬௭௮௯",
    "_type": "numeric"
  },
  "osma": {
    "_digits": "0௧௨௩௪௫௬௭௮௯",
    "_type": "numeric"
  },
  "roman": {
    "_rules": "roman-upper",
    "_type": "algorithmic"
  },
  "romanlow": {
    "_rules": "roman-lower",
    "_type": "algorithmic"
  },
  "saur": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "shrd": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "sind": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "sinh": {
    "_digits": "⁂௧௨௩௪௫௬௭௮௯",
    "_type": "numeric"
  },
  "sora": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "sund": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "takr": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "taluk": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "tamil": {
    "_rules": "tamil",
    "_type": "algorithmic"
  },
  "tamildec": {
    "_digits": "0௧௨௩௪௫௬௭௮௯",
    "_type": "numeric"
  },

```

```

    "telu": {
        "_digits": "୦୧୨୩୪୫୬୭୮୯",
        "_type": "numeric"
    },
    "thai": {
        "_digits": "๐๑๒๓๔๕๖๗๘๙",
        "_type": "numeric"
    },
    "tibet": {
        "_digits": "༠༡༢༣༤༥༦༧༨༩",
        "_type": "numeric"
    },
    "tirh": {
        "_digits": "୦୧୨୩୪୫୬୭୮୯",
        "_type": "numeric"
    },
    "vaii": {
        "_digits": "𐌐𐌑𐌒𐌓𐌔𐌕𐌖𐌗𐌘𐌙",
        "_type": "numeric"
    },
    "wara": {
        "_digits": "୦୧୨୩୪୫୬୭୮୯",
        "_type": "numeric"
    }
}

```

NUMBERINGSYSTEMS.JSX

```
{
    "supplemental";
    {
        "version";
        {
            "_number";
            "$Revision: 12732 $",
            "_unicodeVersion";
            "9.0.0",
            "_cldrVersion";
            "31";
        }
        "numberingSystems";
        {
            "adlm";
            {
                "_digits";
                "୧୪୫୬୭୮୯୦",
                "_type";
                "numeric";
            }
            "ahom";
            {
                "_digits";
                "ꠄꠅ꠆ꠇꠈꠉꠊꠋꠌꠍꠎꠏꠐꠑꠒꠓꠔꠕꠖꠗꠘꠙꠚꠛꠜꠝꠞꠟꠠꠡꠢꠣꠤꠥꠦꠧ꠨꠩꠰꠱꠲꠳꠴꠵꠶꠷꠸꠹꠺꠻꠼꠽꠾꠿ꡀꡁꡂꡃꡄꡅꡆꡇꡈꡉꡐꡑꡒꡓꡔꡕꡖꡗꡘꡙꡚꡛꡜꡝꡞꡟꡠꡡꡢꡣꡤꡥꡦꡧꡨꡩꡰꡱꡲꡳ꡴꡵꡶꡷꡸꡹꡺꡻꡼꡽꡾꡿ꢀꢁꢂꢃꢄꢅꢆꢇꢈꢉꢐꢑꢒꢓꢔꢕꢖꢗꢘꢙꢚꢛꢜꢝꢞꢟꢠꢡꢢꢣꢤꢥꢦꢧꢨꢩꢰꢱꢲꢳꢴꢵꢶꢷꢸꢹꢺꢻꢼꢽꢾꢿꣀꣁꣂꣃ꣄ꣅ꣆꣇꣈꣉꣐꣑꣒꣓꣔꣕꣖꣗꣘꣙꣚꣛꣜꣝꣞꣟꣠꣡꣢꣣꣤꣥꣦꣧꣨꣩꣰꣱ꣲꣳꣴꣵꣶꣷ꣸꣹꣺ꣻ꣼ꣽꣾꣿ꤀꤁꤂꤃꤄꤅꤆꤇꤈꤉ꤐꤑꤒꤓꤔꤕꤖꤗꤘꤙꤚꤛꤜꤝꤞꤟꤠꤡꤢꤣꤤꤥꤦꤧꤨꤩꤰꤱꤲꤳꤴꤵꤶꤷꤸꤹꤺꤻꤼꤽꤾꤿꥀꥁꥂꥃꥄꥅꥆꥇꥈꥉꥐꥑꥒ꥓꥔꥕꥖꥗꥘꥙꥚꥛꥜꥝꥞꥟ꥠꥡꥢꥣꥤꥥꥦꥧꥨꥩꥰꥱꥲꥳꥴꥵꥶꥷꥸꥹꥺꥻꥼ꥽꥾꥿ꦀꦁꦂꦃꦄꦅꦆꦇꦈꦉꦐꦑꦒꦓꦔꦕꦖꦗꦘꦙꦚꦛꦜꦝꦞꦟꦠꦡꦢꦣꦤꦥꦦꦧꦨꦩꦪꦫꦬꦭꦮꦯꦰꦱꦲ꦳ꦴꦵꦶꦷꦸꦹꦺꦻꦼꦽꦾꦿ꧀꧁꧂꧃꧄꧅꧆꧇꧈꧉꧐꧑꧒꧓꧔꧕꧖꧗꧘꧙꧚꧛꧜꧝꧞꧟ꧠꧡꧢꧣꧤꧥꧦꧧꧨꧩ꧰꧱꧲꧳꧴꧵꧶꧷꧸꧹ꧺꧻꧼꧽꧾ꧿ꨀꨁꨂꨃꨄꨅꨆꨇꨈꨉꨐꨑꨒꨓꨔꨕꨖꨗꨘꨙꨚꨛꨜꨝꨞꨟꨠꨡꨢꨣꨤꨥꨦꨧꨨꨩꨰꨱꨲꨳꨴꨵꨶ꨷꨸꨹꨺꨻꨼꨽꨾꨿ꩀꩁꩂꩃꩄꩅꩆꩇꩈꩉ꩐꩑꩒꩓꩔꩕꩖꩗꩘꩙꩚꩛꩜꩝꩞꩟ꩠꩡꩢꩣꩤꩥꩦꩧꩨꩩꩰꩱꩲꩳꩴꩵꩶ꩷꩸꩹ꩺꩻꩼꩽꩾꩿꪀꪁꪂꪃꪄꪅꪆꪇꪈꪉꪐꪑꪒꪓꪔꪕꪖꪗꪘꪙꪚꪛꪜꪝꪞꪟꪠꪡꪢꪣꪤꪥꪦꪧꪨꪩꪰꪱꪴꪲꪳꪵꪶꪷꪸꪹꪺꪻꪼꪽꪾ꪿ꫀ꫁ꫂ꫃꫄꫅꫆꫇꫈꫉꫐꫑꫒꫓꫔꫕꫖꫗꫘꫙꫚ꫛꫜꫝ꫞꫟ꫠꫡꫢꫣꫤꫥꫦꫧꫨꫩ꫰꫱ꫲꫳꫴꫵ꫶꫷꫸꫹꫺꫻꫼꫽꫾꫿꬀ꬁꬂꬃꬄꬅꬆ꬇꬈ꬉ꬐ꬑꬒꬓꬔꬕꬖ꬗꬘꬙꬚꬛꬜꬝꬞꬟ꬠꬡꬢꬣꬤꬥꬦ꬧ꬨꬩꬰꬱꬲꬳꬴꬵꬶꬷꬸꬹꬺꬻꬼꬽꬾꬿꭀꭁꭂꭃꭄꭅꭆꭇꭈꭉꭐꭑꭒꭓꭔꭕꭖꭗꭘꭙꭚ꭛ꭜꭝꭞꭟꭠꭡꭢꭣꭤꭥꭦꭧꭨꭩꭰꭱꭲꭳꭴꭵꭶꭷꭸꭹꭺꭻꭼꭽꭾꭿꮀꮁꮂꮃꮄꮅꮆꮇꮈꮉꮐꮑꮒꮓꮔꮕꮖꮗꮘꮙꮚꮛꮜꮝꮞꮟꮠꮡꮢꮣꮤꮥꮦꮧꮨꮩꮰꮱꮲꮳꮴꮵꮶꮷꮸꮹꮺꮻꮼꮽꮾꮿꯀꯁꯂꯃꯄꯅꯆꯇꯈꯉꯐꯑꯒꯓꯔꯕꯖꯗꯘꯙꯚꯛꯜꯝꯞꯟꯠꯡꯢꯣꯤꯥꯦꯧꯨꯩ꯰꯱꯲꯳꯴꯵꯶꯷꯸꯹꯺꯻꯼꯽꯾꯿가각갂갃간갅갆갇갈갉감갑값갓갔강갖갗갘같갚갛개객갞갟갠갡갢갣갤갥갦갧갨갩갰갱갲갳갴갵갶갷갸갹갺갻갼갽갾갿걀걁걂걃걄걅걆걇걈걉걐걑걒걓걔걕걖걗걘걙걚걛걜걝걞걟걠걡걢걣걤걥걦걧걨걩거걱걲걳건걵걶걷걸걹걺걻걼걽걾걿검겁겂것겄겅겆겇겈겉겐겑겒겓겔겕겖겗겘겙겚겛겜겝겞겟겠겡겢겣겤겥겦겧겨격결겱겲겳겴겵겶겷겸겹겺겻겼경겾겿곀곁곂곃계곅곆곇곈곉곐곑곒곓곔곕곖곗곘곙곚곛곜곝곞곟고곡곢곣곤곥곦곧골곩곰곱곲곳곴공곶곷곸곹곺곻과곽곾곿관괁괂괃괄괅괆괇괈괉괐광괒괓괔괕괖괗괘괙괚괛괜괝괞괟괠괡괢괣괤괥괦괧괨괩괰괱괲괳괴괵괶괷괸괹괺괻괼괽괾괿굀굁굂굃굄굅굆굇굈굉교굑굒굓굔굕굖굗굘굙굚굛굜굝굞굟굠굡굢굣굤굥굦굧굨굩군굱굲굳굴굵굶굷굸굹굺굻굼굽굾굿궀궁궂궃궄궅궆궇궈궉궐궑궒궓궔궕궖궗궘궙궚궛궜궝궞궟궠궡궢궣궤궥궦궧궨궩궰궱궲궳궴궵궶궷궸궹궺궻궼궽궾궿귀귁귂귃귄귅귆귇귈귉귐귑귒귓귔귕귖귗귘귙귚귛규귝귞귟균귡귢귣귤귥귦귧귨귩귰귱귲귳귴귵귶귷그극귺귻근귽귾귿글긁긂긃긄긅긆긇금급긐긑긒긓긔긕긖긗긘긙긚긛긜긝긞긟긠긡긢긣긤긥긦긧긨긩기긱긲긳긴긵긶긷길긹긺긻긼긽긾긿김깁깂깃깄깅깆깇깈깉
```



```

        "_type";
        "numeric";
    }
    "cakm";
    {
        "_digits";
        "ᐃᐅᐃᐅᐅᐅᐅᐅᐅᐅᐅ",
        "_type";
        "numeric";
    }
    "cham";
    {
        "_digits";
        "ᐁᐁᐁᐁᐁᐁᐁᐁᐁᐁ",
        "_type";
        "numeric";
    }
    "cyrl";
    {
        "_rules";
        "cyrillic-lower",
        "_type";
        "algorithmic";
    }
    "deva";
    {
        "_digits";
        "ॐ२३४५६७८९",
        "_type";
        "numeric";
    }
    "ethi";
    {
        "_rules";
        "ethiopic",
        "_type";
        "algorithmic";
    }
    "fullwide";
    {
        "_digits";
        "ᐁ ᠐ 1 2 3 4 5 6 7 8 9",
        "_type";
        "numeric";
    }
    "geor";
    {
        "_rules";
        "georgian",
        "_type";
        "algorithmic";
    }
    "grek";
    {
        "_rules";
        "greek-upper",

```

```

        "_type";
        "algorithmic";
    }
    "greklow";
    {
        "_rules";
        "greek-lower",
        "_type";
        "algorithmic";
    }
    "gujr";
    {
        "_digits";
        "૦૧૨૩૪૫૬૭૮૯",
        "_type";
        "numeric";
    }
    "guru";
    {
        "_digits";
        "੦੧੨੩੪੫੬੭੮੯",
        "_type";
        "numeric";
    }
    "hanidays";
    {
        "_rules";
        "zh/SpelloutRules/spellout-numbering-days",
        "_type";
        "algorithmic";
    }
    "hanidec";
    {
        "_digits";
        "〇一二三四五六七八九",
        "_type";
        "numeric";
    }
    "hans";
    {
        "_rules";
        "zh/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "hansfin";
    {
        "_rules";
        "zh/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "hant";
    {
        "_rules";
        "zh_Hant/SpelloutRules/spellout-cardinal",

```

```

        "_type";
        "algorithmic";
    }
    "hantfin";
    {
        "_rules";
        "zh_Hant/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "hebr";
    {
        "_rules";
        "hebrew",
        "_type";
        "algorithmic";
    }
    "hmng";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
    "java";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉၁၀၁၁",
        "_type";
        "numeric";
    }
    "jpan";
    {
        "_rules";
        "ja/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "jpanfin";
    {
        "_rules";
        "ja/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "kali";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
    "khmr";
    {
        "_digits";

```



```

        "၀၅၉၈၇၆၅၄၃၂၁",
        "_type";
        "numeric";
    }
    "knda";
    {
        "_digits";
        "೦೧೨೩೪೫೬೭೮೯",
        "_type";
        "numeric";
    }
    "lana";
    {
        "_digits";
        "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉",
        "_type";
        "numeric";
    }
    "lanatham";
    {
        "_digits";
        "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉",
        "_type";
        "numeric";
    }
    "laoo";
    {
        "_digits";
        "ᦀᦁᦂᦃᦄᦅᦆᦇᦈᦉ",
        "_type";
        "numeric";
    }
    "latn";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "lepc";
    {
        "_digits";
        "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉",
        "_type";
        "numeric";
    }
    "limb";
    {
        "_digits";
        "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉",
        "_type";
        "numeric";
    }
    "mathbold";
    {
        "_digits";

```

```

        "0123456789",
        "_type";
        "numeric";
    }
    "mathdbl";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathmono";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathsanb";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathsans";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mlym";
    {
        "_digits";
        "ഐഹനരഭിനാഗവുൻ",
        "_type";
        "numeric";
    }
    "modi";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
    "mong";
    {
        "_digits";
        "᠐᠑᠒᠐ᠠᠨᠵᠢᠨᠠᠨᠠᠨᠠᠨ",
        "_type";
        "numeric";
    }
    "mroo";
    {
        "_digits";

```

```

        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "mtei";
    {
        "_digits";
        "ၵၵၵၵၵၵၵၵ",
        "_type";
        "numeric";
    }
    "mymr";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈",
        "_type";
        "numeric";
    }
    "mymrshan";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈",
        "_type";
        "numeric";
    }
    "mymrtlng";
    {
        "_digits";
        "ၵၵၵၵၵၵၵၵ",
        "_type";
        "numeric";
    }
    "newa";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "nkoo";
    {
        "_digits";
        "ၵၵၵၵၵၵၵၵ",
        "_type";
        "numeric";
    }
    "olck";
    {
        "_digits";
        "ၵၵၵၵၵၵၵၵ",
        "_type";
        "numeric";
    }
    "orya";

```

```

    {
      "_digits";
      "௦௧௨௩௪௫௬௭௮௯",
      "_type";
      "numeric";
    }
    "osma";
    {
      "_digits";
      "௦௧௨௩௪௫௬௭௮௯",
      "_type";
      "numeric";
    }
    "roman";
    {
      "_rules";
      "roman-upper",
      "_type";
      "algorithmic";
    }
    "romanlow";
    {
      "_rules";
      "roman-lower",
      "_type";
      "algorithmic";
    }
    "saur";
    {
      "_digits";
      "௦௧௨௩௪௫௬௭௮௯",
      "_type";
      "numeric";
    }
    "shrd";
    {
      "_digits";
      "௦௧௨௩௪௫௬௭௮௯",
      "_type";
      "numeric";
    }
    "sind";
    {
      "_digits";
      "௦௧௨௩௪௫௬௭௮௯",
      "_type";
      "numeric";
    }
    "sinh";
    {
      "_digits";
      "௦௧௨௩௪௫௬௭௮௯",
      "_type";
      "numeric";
    }
    "sora";

```

```

    {
      "_digits";
      "0123456789",
      "_type";
      "numeric";
    }
    "sund";
    {
      "_digits";
      "0123456789",
      "_type";
      "numeric";
    }
    "takr";
    {
      "_digits";
      "0123456789",
      "_type";
      "numeric";
    }
    "talu";
    {
      "_digits";
      "0123456789",
      "_type";
      "numeric";
    }
    "taml";
    {
      "_rules";
      "tamil",
      "_type";
      "algorithmic";
    }
    "tamldec";
    {
      "_digits";
      "0123456789",
      "_type";
      "numeric";
    }
    "telu";
    {
      "_digits";
      "0123456789",
      "_type";
      "numeric";
    }
    "thai";
    {
      "_digits";
      "0123456789",
      "_type";
      "numeric";
    }
    "tib";

```

[illegible]

NUMBERS.JSON

```
{
  "main": {
    "ar": {
      "identity": {
        "version": {
          "_number": "$Revision: 12879 $",
          "_cldrVersion": "30.0.3"
        },
        "language": "ar"
      },
      "numbers": {
        "defaultNumberingSystem": "arab",
        "otherNumberingSystems": {
          "native": "arab"
        },
        "minimumGroupingDigits": "1",
        "symbols-numberSystem-arab": {
          "decimal": ",",
          "group": ",",
          "list": ":",
          "percentSign": "%",
          "plusSign": "+",
          "minusSign": "-",

```

```

    "exponential": "أس",
    "superscriptingExponent": "×",
    "perMille": "%",
    "infinity": "∞",
    "nan": "ليس رقم",
    "timeSeparator": ":"
  },
  "symbols-numberSystem-latn": {
    "decimal": ".",
    "group": ",",
    "list": ";",
    "percentSign": "%",
    "plusSign": "+",
    "minusSign": "-",
    "exponential": "E",
    "superscriptingExponent": "×",
    "perMille": "‰",
    "infinity": "∞",
    "nan": "ليس رقمًا",
    "timeSeparator": ":"
  },
  "decimalFormats-numberSystem-arab": {
    "standard": "#,##0.###",
    "long": {
      "decimalFormat": {
        "1000-count-zero": "0 ألف",
        "1000-count-one": "0 ألف",
        "1000-count-two": "0 ألف",
        "1000-count-few": "0 آلاف",
        "1000-count-many": "0 ألف",
        "1000-count-other": "0 ألف",
        "10000-count-zero": "00 ألف",
        "10000-count-one": "00 ألف",
        "10000-count-two": "00 ألف",
        "10000-count-few": "00 ألف",
        "10000-count-many": "00 ألف",
        "10000-count-other": "00 ألف",
        "100000-count-zero": "000 ألف",
        "100000-count-one": "000 ألف",
        "100000-count-two": "000 ألف",
        "100000-count-few": "000 ألف",
        "100000-count-many": "000 ألف",
        "100000-count-other": "000 ألف",
        "1000000-count-zero": "0 مليون",
        "1000000-count-one": "0 مليون",
        "1000000-count-two": "0 مليون",
        "1000000-count-few": "0 ملايين",
        "1000000-count-many": "0 مليون",
        "1000000-count-other": "0 مليون",
        "10000000-count-zero": "00 مليون",
        "10000000-count-one": "00 مليون",
        "10000000-count-two": "00 مليون",
        "10000000-count-few": "00 ملايين",
        "10000000-count-many": "00 مليون",
        "10000000-count-other": "00 مليون",
        "100000000-count-zero": "000 مليون",
        "100000000-count-one": "000 مليون",

```

```

        "1000000000-count-two": "000 مليون",
        "1000000000-count-few": "000 مليون",
        "1000000000-count-many": "000 مليون",
        "1000000000-count-other": "000 مليون",
        "1000000000-count-zero": "0 مليار",
        "1000000000-count-one": "0 مليار",
        "1000000000-count-two": "0 مليار",
        "1000000000-count-few": "0 مليار",
        "1000000000-count-many": "0 مليار",
        "1000000000-count-other": "0 مليار",
        "1000000000-count-zero": "00 مليار",
        "1000000000-count-one": "00 مليار",
        "1000000000-count-two": "00 مليار",
        "1000000000-count-few": "00 مليار",
        "1000000000-count-many": "00 مليار",
        "1000000000-count-other": "00 مليار",
        "10000000000-count-zero": "000 مليار",
        "10000000000-count-one": "000 مليار",
        "10000000000-count-two": "000 مليار",
        "10000000000-count-few": "000 مليار",
        "10000000000-count-many": "000 مليار",
        "10000000000-count-other": "000 مليار",
        "1000000000000-count-zero": "0 تريليون",
        "1000000000000-count-one": "0 تريليون",
        "1000000000000-count-two": "0 تريليون",
        "1000000000000-count-few": "0 تريليونات",
        "1000000000000-count-many": "0 تريليون",
        "1000000000000-count-other": "0 تريليون",
        "10000000000000-count-zero": "00 تريليون",
        "10000000000000-count-one": "00 تريليون",
        "10000000000000-count-two": "00 تريليون",
        "10000000000000-count-few": "00 تريليون",
        "10000000000000-count-many": "00 تريليون",
        "10000000000000-count-other": "00 تريليون",
        "100000000000000-count-zero": "000 تريليون",
        "100000000000000-count-one": "000 تريليون",
        "100000000000000-count-two": "000 تريليون",
        "100000000000000-count-few": "000 تريليون",
        "100000000000000-count-many": "000 تريليون",
        "100000000000000-count-other": "000 تريليون"
    },
    },
    "short": {
        "decimalFormat": {
            "1000-count-zero": "0 ألف",
            "1000-count-one": "0 ألف",
            "1000-count-two": "0 ألف",
            "1000-count-few": "0 آلاف",
            "1000-count-many": "0 ألف",
            "1000-count-other": "0 ألف",
            "10000-count-zero": "00 ألف",
            "10000-count-one": "00 ألف",
            "10000-count-two": "00 ألف",
            "10000-count-few": "00 ألف",
            "10000-count-many": "00 ألف",
            "10000-count-other": "00 ألف",
            "100000-count-zero": "000 ألف",

```



```
"100000-count-one": "000 ألف",
"100000-count-two": "000 ألف",
"100000-count-few": "000 ألف",
"100000-count-many": "000 ألف",
"100000-count-other": "000 ألف",
"1000000-count-zero": "0 مليون",
"1000000-count-one": "0 مليون",
"1000000-count-two": "0 مليون",
"1000000-count-few": "0 مليون",
"1000000-count-many": "0 مليون",
"1000000-count-other": "0 مليون",
"10000000-count-zero": "00 مليون",
"10000000-count-one": "00 مليون",
"10000000-count-two": "00 مليون",
"10000000-count-few": "00 مليون",
"10000000-count-many": "00 مليون",
"10000000-count-other": "00 مليون",
"100000000-count-zero": "000 مليون",
"100000000-count-one": "000 مليون",
"100000000-count-two": "000 مليون",
"100000000-count-few": "000 مليون",
"100000000-count-many": "000 مليون",
"100000000-count-other": "000 مليون",
"1000000000-count-zero": "0 مليا",
"1000000000-count-one": "0 مليا",
"1000000000-count-two": "0 مليا",
"1000000000-count-few": "0 مليا",
"1000000000-count-many": "0 مليا",
"1000000000-count-other": "0 مليا",
"10000000000-count-zero": "00 مليا",
"10000000000-count-one": "00 مليا",
"10000000000-count-two": "00 مليا",
"10000000000-count-few": "00 مليا",
"10000000000-count-many": "00 مليا",
"10000000000-count-other": "00 مليا",
"100000000000-count-zero": "000 مليا",
"100000000000-count-one": "000 مليا",
"100000000000-count-two": "000 مليا",
"100000000000-count-few": "000 مليا",
"100000000000-count-many": "000 مليا",
"100000000000-count-other": "000 مليا",
"1000000000000-count-zero": "0 ترليو",
"1000000000000-count-one": "0 ترليو",
"1000000000000-count-two": "0 ترليو",
"1000000000000-count-few": "0 ترليو",
"1000000000000-count-many": "0 ترليو",
"1000000000000-count-other": "0 ترليو",
"10000000000000-count-zero": "00 ترليو",
"10000000000000-count-one": "00 ترليو",
"10000000000000-count-two": "00 ترليو",
"10000000000000-count-few": "00 ترليو",
"10000000000000-count-many": "00 ترليو",
"10000000000000-count-other": "00 ترليو",
"100000000000000-count-zero": "000 ترليو",
"100000000000000-count-one": "000 ترليو",
"100000000000000-count-two": "000 ترليو",
"100000000000000-count-few": "000 ترليو",
```

```

        "1000000000000000-count-many": "000 ترليو",
        "1000000000000000-count-other": "000 ترليو"
    }
}
},
"decimalFormats-numberSystem-latn": {
    "standard": "#,##0.###",
    "long": {
        "decimalFormat": {
            "1000-count-zero": "0 ألف",
            "1000-count-one": "0 ألف",
            "1000-count-two": "0 ألف",
            "1000-count-few": "0 آلاف",
            "1000-count-many": "0 ألف",
            "1000-count-other": "0 ألف",
            "10000-count-zero": "00 ألف",
            "10000-count-one": "00 ألف",
            "10000-count-two": "00 ألف",
            "10000-count-few": "00 ألف",
            "10000-count-many": "00 ألف",
            "10000-count-other": "00 ألف",
            "100000-count-zero": "000 ألف",
            "100000-count-one": "000 ألف",
            "100000-count-two": "000 ألف",
            "100000-count-few": "000 ألف",
            "100000-count-many": "000 ألف",
            "100000-count-other": "000 ألف",
            "1000000-count-zero": "0 مليون",
            "1000000-count-one": "0 مليون",
            "1000000-count-two": "0 مليون",
            "1000000-count-few": "0 ملايين",
            "1000000-count-many": "0 مليون",
            "1000000-count-other": "0 مليون",
            "10000000-count-zero": "00 مليون",
            "10000000-count-one": "00 مليون",
            "10000000-count-two": "00 مليون",
            "10000000-count-few": "00 ملايين",
            "10000000-count-many": "00 مليون",
            "10000000-count-other": "00 مليون",
            "100000000-count-zero": "000 مليون",
            "100000000-count-one": "000 مليون",
            "100000000-count-two": "000 مليون",
            "100000000-count-few": "000 مليون",
            "100000000-count-many": "000 مليون",
            "100000000-count-other": "000 مليون",
            "1000000000-count-zero": "0 مليار",
            "1000000000-count-one": "0 مليار",
            "1000000000-count-two": "0 مليار",
            "1000000000-count-few": "0 مليارات",
            "1000000000-count-many": "0 مليار",
            "1000000000-count-other": "0 مليار",
            "10000000000-count-zero": "00 مليار",
            "10000000000-count-one": "00 مليار",
            "10000000000-count-two": "00 مليار",
            "10000000000-count-few": "00 مليارات",
            "10000000000-count-many": "00 مليار",
            "10000000000-count-other": "00 مليار",

```

```

        "100000000000-count-zero": "000 مليار",
        "100000000000-count-one": "000 مليار",
        "100000000000-count-two": "000 مليار",
        "100000000000-count-few": "000 مليار",
        "100000000000-count-many": "000 مليار",
        "100000000000-count-other": "000 مليار",
        "100000000000-count-zero": "0 تريليون",
        "100000000000-count-one": "0 تريليون",
        "100000000000-count-two": "0 تريليون",
        "100000000000-count-few": "0 تريليونات",
        "100000000000-count-many": "0 تريليون",
        "100000000000-count-other": "0 تريليون",
        "1000000000000-count-zero": "00 تريليون",
        "1000000000000-count-one": "00 تريليون",
        "1000000000000-count-two": "00 تريليون",
        "1000000000000-count-few": "00 تريليون",
        "1000000000000-count-many": "00 تريليون",
        "1000000000000-count-other": "00 تريليون",
        "10000000000000-count-zero": "000 تريليون",
        "10000000000000-count-one": "000 تريليون",
        "10000000000000-count-two": "000 تريليون",
        "10000000000000-count-few": "000 تريليون",
        "10000000000000-count-many": "000 تريليون",
        "10000000000000-count-other": "000 تريليون"
    }
},
"short": {
    "decimalFormat": {
        "1000-count-zero": "0 ألف",
        "1000-count-one": "0 ألف",
        "1000-count-two": "0 ألف",
        "1000-count-few": "0 آلاف",
        "1000-count-many": "0 ألف",
        "1000-count-other": "0 ألف",
        "10000-count-zero": "00 ألف",
        "10000-count-one": "00 ألف",
        "10000-count-two": "00 ألف",
        "10000-count-few": "00 ألف",
        "10000-count-many": "00 ألف",
        "10000-count-other": "00 ألف",
        "100000-count-zero": "000 ألف",
        "100000-count-one": "000 ألف",
        "100000-count-two": "000 ألف",
        "100000-count-few": "000 ألف",
        "100000-count-many": "000 ألف",
        "100000-count-other": "000 ألف",
        "1000000-count-zero": "0 مليون",
        "1000000-count-one": "0 مليون",
        "1000000-count-two": "0 مليون",
        "1000000-count-few": "0 مليون",
        "1000000-count-many": "0 مليون",
        "1000000-count-other": "0 مليون",
        "10000000-count-zero": "00 مليون",
        "10000000-count-one": "00 مليون",
        "10000000-count-two": "00 مليون",
        "10000000-count-few": "00 مليون",
        "10000000-count-many": "00 مليون",
    }
}

```

```

        "100000000-count-other": "00 مليو",
        "100000000-count-zero": "000 مليو",
        "100000000-count-one": "000 مليو",
        "100000000-count-two": "000 مليو",
        "100000000-count-few": "000 مليو",
        "100000000-count-many": "000 مليو",
        "100000000-count-other": "000 مليو",
        "1000000000-count-zero": "0 مليا",
        "1000000000-count-one": "0 مليا",
        "1000000000-count-two": "0 مليا",
        "1000000000-count-few": "0 مليا",
        "1000000000-count-many": "0 مليا",
        "1000000000-count-other": "0 مليا",
        "10000000000-count-zero": "00 مليا",
        "10000000000-count-one": "00 مليا",
        "10000000000-count-two": "00 مليا",
        "10000000000-count-few": "00 مليا",
        "10000000000-count-many": "00 مليا",
        "10000000000-count-other": "00 مليا",
        "100000000000-count-zero": "000 مليا",
        "100000000000-count-one": "000 مليا",
        "100000000000-count-two": "000 مليا",
        "100000000000-count-few": "000 مليا",
        "100000000000-count-many": "000 مليا",
        "100000000000-count-other": "000 مليا",
        "1000000000000-count-zero": "0 ترليو",
        "1000000000000-count-one": "0 ترليو",
        "1000000000000-count-two": "0 ترليو",
        "1000000000000-count-few": "0 ترليو",
        "1000000000000-count-many": "0 ترليو",
        "1000000000000-count-other": "0 ترليو",
        "10000000000000-count-zero": "00 ترليو",
        "10000000000000-count-one": "00 ترليو",
        "10000000000000-count-two": "00 ترليو",
        "10000000000000-count-few": "00 ترليو",
        "10000000000000-count-many": "00 ترليو",
        "10000000000000-count-other": "00 ترليو",
        "100000000000000-count-zero": "000 ترليو",
        "100000000000000-count-one": "000 ترليو",
        "100000000000000-count-two": "000 ترليو",
        "100000000000000-count-few": "000 ترليو",
        "100000000000000-count-many": "000 ترليو",
        "100000000000000-count-other": "000 ترليو"
    }
}
},
"scientificFormats-numberSystem-arab": {
    "standard": "#E0"
},
"scientificFormats-numberSystem-latn": {
    "standard": "#E0"
},
"percentFormats-numberSystem-arab": {
    "standard": "#,##0 %"
},
"percentFormats-numberSystem-latn": {
    "standard": "#,##0%"
}

```

```

    },
    "currencyFormats-numberSystem-arab": {
      "currencySpacing": {
        "beforeCurrency": {
          "currencyMatch": "[:^S:]",
          "surroundingMatch": "[:digit:]",
          "insertBetween": " "
        },
        "afterCurrency": {
          "currencyMatch": "[:^S:]",
          "surroundingMatch": "[:digit:]",
          "insertBetween": " "
        }
      },
      "standard": "#,##0.00 ¤",
      "accounting": "#,##0.00 ¤",
      "unitPattern-count-zero": "{0} {1}",
      "unitPattern-count-one": "{0} {1}",
      "unitPattern-count-two": "{0} {1}",
      "unitPattern-count-few": "{0} {1}",
      "unitPattern-count-many": "{0} {1}",
      "unitPattern-count-other": "{0} {1}"
    },
    "currencyFormats-numberSystem-latn": {
      "currencySpacing": {
        "beforeCurrency": {
          "currencyMatch": "[:^S:]",
          "surroundingMatch": "[:digit:]",
          "insertBetween": " "
        },
        "afterCurrency": {
          "currencyMatch": "[:^S:]",
          "surroundingMatch": "[:digit:]",
          "insertBetween": " "
        }
      },
      "standard": "¤ #,##0.00",
      "accounting": "¤#,##0.00; (¤#,##0.00)",
      "short": {
        "standard": {
          "1000-count-zero": "¤ 0 ألف",
          "1000-count-one": "¤ 0 ألف",
          "1000-count-two": "¤ 0 ألف",
          "1000-count-few": "¤ 0 ألف",
          "1000-count-many": "¤ 0 ألف",
          "1000-count-other": "¤ 0 ألف",
          "10000-count-zero": "¤ 00 ألف",
          "10000-count-one": "¤ 00 ألف",
          "10000-count-two": "¤ 00 ألف",
          "10000-count-few": "¤ 00 ألف",
          "10000-count-many": "¤ 00 ألف",
          "10000-count-other": "¤ 00 ألف",
          "100000-count-zero": "¤ 000 ألف",
          "100000-count-one": "¤ 000 ألف",
          "100000-count-two": "¤ 000 ألف",
          "100000-count-few": "¤ 000 ألف",
          "100000-count-many": "¤ 000 ألف",

```

```

"100000-count-other": "ألف 000",
"1000000-count-zero": "مليون 0",
"1000000-count-one": "مليون 0",
"1000000-count-two": "مليون 0",
"1000000-count-few": "مليون 0",
"1000000-count-many": "مليون 0",
"1000000-count-other": "مليون 0",
"10000000-count-zero": "مليون 00",
"10000000-count-one": "مليون 00",
"10000000-count-two": "مليون 00",
"10000000-count-few": "مليون 00",
"10000000-count-many": "مليون 00",
"100000000-count-zero": "مليون 000",
"100000000-count-one": "مليون 000",
"100000000-count-two": "مليون 000",
"100000000-count-few": "مليون 000",
"100000000-count-many": "مليون 000",
"1000000000-count-zero": "ملياً 0",
"1000000000-count-one": "ملياً 0",
"1000000000-count-two": "ملياً 0",
"1000000000-count-few": "ملياً 0",
"1000000000-count-many": "ملياً 0",
"10000000000-count-zero": "ملياً 00",
"10000000000-count-one": "ملياً 00",
"10000000000-count-two": "ملياً 00",
"10000000000-count-few": "ملياً 00",
"10000000000-count-many": "ملياً 00",
"100000000000-count-zero": "ملياً 000",
"100000000000-count-one": "ملياً 000",
"100000000000-count-two": "ملياً 000",
"100000000000-count-few": "ملياً 000",
"100000000000-count-many": "ملياً 000",
"1000000000000-count-zero": "ترليون 0",
"1000000000000-count-one": "ترليون 0",
"1000000000000-count-two": "ترليون 0",
"1000000000000-count-few": "ترليون 0",
"1000000000000-count-many": "ترليون 0",
"10000000000000-count-zero": "ترليون 00",
"10000000000000-count-one": "ترليون 00",
"10000000000000-count-two": "ترليون 00",
"10000000000000-count-few": "ترليون 00",
"100000000000000-count-zero": "ترليون 000",
"100000000000000-count-one": "ترليون 000",
"100000000000000-count-two": "ترليون 000",
"100000000000000-count-few": "ترليون 000",
"1000000000000000-count-zero": "ترليون 000",
"1000000000000000-count-other": "ترليون 000"
    },
  },

```

```

        "unitPattern-count-zero": "{0} {1}",
        "unitPattern-count-one": "{0} {1}",
        "unitPattern-count-two": "{0} {1}",
        "unitPattern-count-few": "{0} {1}",
        "unitPattern-count-many": "{0} {1}",
        "unitPattern-count-other": "{0} {1}"
    },
    "miscPatterns-numberSystem-arab": {
        "atLeast": "+{0}",
        "range": "{0}-{1}"
    },
    "miscPatterns-numberSystem-latn": {
        "atLeast": "+{0}",
        "range": "{0}-{1}"
    }
}
}
}
}
}

```

NUMBERS.JSX

```

{
    "main";
    {
        "ar";
        {
            "identity";
            {
                "version";
                {
                    "_number";
                    "$Revision: 12879 $",
                    "_cldrVersion";
                    "30.0.3";
                }
                "language";
                "ar";
            }
            "numbers";
            {
                "defaultNumberingSystem";
                "arab",
                "otherNumberingSystems";
                {
                    "native";
                    "arab";
                }
                "minimumGroupingDigits";
                "1",
                "symbols-numberSystem-arab";
                {
                    "decimal";
                    ",",
                    "group";
                    ",",

```

```

        "list";
        "؛",
        "percentSign";
        "%",
        "plusSign";
        "+",
        "minusSign";
        "-",
        "exponential";
        "اُس",
        "superscriptingExponent";
        "×",
        "perMille";
        "‰",
        "infinity";
        "∞",
        "nan";
        "ليس رقم",
        "timeSeparator";
        ":";
    }
    "symbols-numberSystem-latn";
    {
        "decimal";
        ".",
        "group";
        ",",
        "list";
        ";",
        "percentSign";
        "%",
        "plusSign";
        "+",
        "minusSign";
        "-",
        "exponential";
        "E",
        "superscriptingExponent";
        "×",
        "perMille";
        "‰",
        "infinity";
        "∞",
        "nan";
        "ليس رقمًا",
        "timeSeparator";
        ":";
    }
    "decimalFormats-numberSystem-arab";
    {
        "standard";
        "#,##0.###",
        "long";
        {
            "decimalFormat";
            {
                "1000-count-zero";
            }
        }
    }

```



```

"0 ألف",
  "1000-count-one";
"0 ألف",
  "1000-count-two";
"0 ألف",
  "1000-count-few";
"0 آلاف",
  "1000-count-many";
"0 ألف",
  "1000-count-other";
"0 ألف",
  "10000-count-zero";
"00 ألف",
  "10000-count-one";
"00 ألف",
  "10000-count-two";
"00 ألف",
  "10000-count-few";
"00 ألف",
  "10000-count-many";
"00 ألف",
  "10000-count-other";
"00 ألف",
  "100000-count-zero";
"000 ألف",
  "100000-count-one";
"000 ألف",
  "100000-count-two";
"000 ألف",
  "100000-count-few";
"000 ألف",
  "100000-count-many";
"000 ألف",
  "100000-count-other";
"000 ألف",
  "1000000-count-zero";
"0 مليون",
  "1000000-count-one";
"0 مليون",
  "1000000-count-two";
"0 مليون",
  "1000000-count-few";
"0 ملايين",
  "1000000-count-many";
"0 مليون",
  "1000000-count-other";
"0 مليون",
  "10000000-count-zero";
"00 مليون",
  "10000000-count-one";
"00 مليون",
  "10000000-count-two";
"00 مليون",
  "10000000-count-few";
"00 ملايين",
  "10000000-count-many";
"00 مليون",

```

```

        "10000000-count-other";
    "00 مليون",
    "100000000-count-zero";
    "000 مليون",
    "100000000-count-one";
    "000 مليون",
    "100000000-count-two";
    "000 مليون",
    "100000000-count-few";
    "000 مليون",
    "100000000-count-many";
    "000 مليون",
    "100000000-count-other";
    "000 مليون",
    "1000000000-count-zero";
    "0 مليار",
    "1000000000-count-one";
    "0 مليار",
    "1000000000-count-two";
    "0 مليار",
    "1000000000-count-few";
    "0 مليار",
    "1000000000-count-many";
    "0 مليار",
    "1000000000-count-other";
    "0 مليار",
    "10000000000-count-zero";
    "00 مليار",
    "10000000000-count-one";
    "00 مليار",
    "10000000000-count-two";
    "00 مليار",
    "10000000000-count-few";
    "00 مليار",
    "10000000000-count-many";
    "00 مليار",
    "10000000000-count-other";
    "00 مليار",
    "100000000000-count-zero";
    "000 مليار",
    "100000000000-count-one";
    "000 مليار",
    "100000000000-count-two";
    "000 مليار",
    "100000000000-count-few";
    "000 مليار",
    "100000000000-count-many";
    "000 مليار",
    "100000000000-count-other";
    "000 مليار",
    "1000000000000-count-zero";
    "0 تربيون",
    "1000000000000-count-one";
    "0 تربيون",
    "1000000000000-count-two";
    "0 تربيون",
    "1000000000000-count-few";

```

```

        "0 تريليونات",
        "1000000000000-count-many";
        "0 تريليون",
        "1000000000000-count-other";
        "0 تريليون",
        "1000000000000-count-zero";
        "00 تريليون",
        "1000000000000-count-one";
        "00 تريليون",
        "1000000000000-count-two";
        "00 تريليون",
        "1000000000000-count-few";
        "00 تريليون",
        "1000000000000-count-many";
        "00 تريليون",
        "1000000000000-count-other";
        "00 تريليون",
        "1000000000000-count-zero";
        "000 تريليون",
        "1000000000000-count-one";
        "000 تريليون",
        "1000000000000-count-two";
        "000 تريليون",
        "1000000000000-count-few";
        "000 تريليون",
        "1000000000000-count-many";
        "000 تريليون",
        "1000000000000-count-other";
        "000 تريليون";
    }
}
"short";
{
    "decimalFormat";
    {
        "1000-count-zero";
        "0 ألف",
        "1000-count-one";
        "0 ألف",
        "1000-count-two";
        "0 ألف",
        "1000-count-few";
        "0 آلاف",
        "1000-count-many";
        "0 ألف",
        "1000-count-other";
        "0 ألف",
        "10000-count-zero";
        "00 ألف",
        "10000-count-one";
        "00 ألف",
        "10000-count-two";
        "00 ألف",
        "10000-count-few";
        "00 ألف",
        "10000-count-many";
        "00 ألف",
    }
}

```

```
"10000-count-other";
"00 ألف",
"100000-count-zero";
"000 ألف",
"100000-count-one";
"000 ألف",
"100000-count-two";
"000 ألف",
"100000-count-few";
"000 ألف",
"100000-count-many";
"000 ألف",
"100000-count-other";
"000 ألف",
"1000000-count-zero";
"0 مليون",
"1000000-count-one";
"0 مليون",
"1000000-count-two";
"0 مليون",
"1000000-count-few";
"0 مليون",
"1000000-count-many";
"0 مليون",
"1000000-count-other";
"0 مليون",
"10000000-count-zero";
"00 مليون",
"10000000-count-one";
"00 مليون",
"10000000-count-two";
"00 مليون",
"10000000-count-few";
"00 مليون",
"10000000-count-many";
"00 مليون",
"10000000-count-other";
"00 مليون",
"100000000-count-zero";
"000 مليون",
"100000000-count-one";
"000 مليون",
"100000000-count-two";
"000 مليون",
"100000000-count-few";
"000 مليون",
"100000000-count-many";
"000 مليون",
"100000000-count-other";
"000 مليون",
"1000000000-count-zero";
"0 مليا",
"1000000000-count-one";
"0 مليا",
"1000000000-count-two";
"0 مليا",
"1000000000-count-few";
```

```
"0 مليا",
  "1000000000-count-many";
"0 مليا",
  "1000000000-count-other";
"0 مليا",
  "1000000000-count-zero";
"00 مليا",
  "1000000000-count-one";
"00 مليا",
  "1000000000-count-two";
"00 مليا",
  "1000000000-count-few";
"00 مليا",
  "1000000000-count-many";
"00 مليا",
  "1000000000-count-other";
"00 مليا",
  "1000000000-count-zero";
"000 مليا",
  "1000000000-count-one";
"000 مليا",
  "1000000000-count-two";
"000 مليا",
  "1000000000-count-few";
"000 مليا",
  "1000000000-count-many";
"000 مليا",
  "1000000000-count-other";
"000 مليا",
  "1000000000-count-zero";
"0 ترليو",
  "100000000000-count-one";
"0 ترليو",
  "100000000000-count-two";
"0 ترليو",
  "100000000000-count-few";
"0 ترليو",
  "100000000000-count-many";
"0 ترليو",
  "100000000000-count-other";
"0 ترليو",
  "100000000000-count-zero";
"00 ترليو",
  "100000000000-count-one";
"00 ترليو",
  "100000000000-count-two";
"00 ترليو",
  "100000000000-count-few";
"00 ترليو",
  "100000000000-count-many";
"00 ترليو",
  "100000000000-count-other";
"00 ترليو",
  "100000000000-count-zero";
"000 ترليو",
  "1000000000000-count-one";
"000 ترليو",
```

```

        "1000000000000000-count-two";
        "000 ترليو",
        "1000000000000000-count-few";
        "000 ترليو",
        "1000000000000000-count-many";
        "000 ترليو",
        "1000000000000000-count-other";
        "000 ترليو";
    }
}
}
"decimalFormats-numberSystem-latn";
{
    "standard";
    "#,##0.###",
    "long";
    {
        "decimalFormat";
        {
            "1000-count-zero";
            "0 ألف",
            "1000-count-one";
            "0 ألف",
            "1000-count-two";
            "0 ألف",
            "1000-count-few";
            "0 آلاف",
            "1000-count-many";
            "0 ألف",
            "1000-count-other";
            "0 ألف",
            "10000-count-zero";
            "00 ألف",
            "10000-count-one";
            "00 ألف",
            "10000-count-two";
            "00 ألف",
            "10000-count-few";
            "00 ألف",
            "10000-count-many";
            "00 ألف",
            "10000-count-other";
            "00 ألف",
            "100000-count-zero";
            "000 ألف",
            "100000-count-one";
            "000 ألف",
            "100000-count-two";
            "000 ألف",
            "100000-count-few";
            "000 ألف",
            "100000-count-many";
            "000 ألف",
            "100000-count-other";
            "000 ألف",
            "1000000-count-zero";
            "0 مليون",

```

```

        "1000000-count-one";
    "0 مليون",
        "1000000-count-two";
    "0 مليون",
        "1000000-count-few";
    "0 ملايين",
        "1000000-count-many";
    "0 مليون",
        "1000000-count-other";
    "0 مليون",
        "10000000-count-zero";
    "00 مليون",
        "10000000-count-one";
    "00 مليون",
        "10000000-count-two";
    "00 مليون",
        "10000000-count-few";
    "00 ملايين",
        "10000000-count-many";
    "00 مليون",
        "10000000-count-other";
    "00 مليون",
        "100000000-count-zero";
    "000 مليون",
        "100000000-count-one";
    "000 مليون",
        "100000000-count-two";
    "000 مليون",
        "100000000-count-few";
    "000 مليون",
        "100000000-count-many";
    "000 مليون",
        "100000000-count-other";
    "000 مليون",
        "1000000000-count-zero";
    "0 مليار",
        "1000000000-count-one";
    "0 مليار",
        "1000000000-count-two";
    "0 مليار",
        "1000000000-count-few";
    "0 مليار",
        "1000000000-count-many";
    "0 مليار",
        "1000000000-count-other";
    "0 مليار",
        "10000000000-count-zero";
    "00 مليار",
        "10000000000-count-one";
    "00 مليار",
        "10000000000-count-two";
    "00 مليار",
        "10000000000-count-few";
    "00 مليار",
        "10000000000-count-many";
    "00 مليار",
        "10000000000-count-other";

```

```

        "00 مليار",
        "100000000000-count-zero";
        "000 مليار",
        "100000000000-count-one";
        "000 مليار",
        "100000000000-count-two";
        "000 مليار",
        "100000000000-count-few";
        "000 مليار",
        "100000000000-count-many";
        "000 مليار",
        "100000000000-count-other";
        "000 مليار",
        "100000000000-count-zero";
        "0 تريليون",
        "1000000000000-count-one";
        "0 تريليون",
        "1000000000000-count-two";
        "0 تريليون",
        "1000000000000-count-few";
        "0 تريليونات",
        "1000000000000-count-many";
        "0 تريليون",
        "1000000000000-count-other";
        "0 تريليون",
        "1000000000000-count-zero";
        "00 تريليون",
        "10000000000000-count-one";
        "00 تريليون",
        "10000000000000-count-two";
        "00 تريليون",
        "10000000000000-count-few";
        "00 تريليون",
        "10000000000000-count-many";
        "00 تريليون",
        "10000000000000-count-other";
        "00 تريليون",
        "100000000000000-count-zero";
        "000 تريليون",
        "100000000000000-count-one";
        "000 تريليون",
        "100000000000000-count-two";
        "000 تريليون",
        "100000000000000-count-few";
        "000 تريليون",
        "100000000000000-count-many";
        "000 تريليون",
        "100000000000000-count-other";
        "000 تريليون";
    }
}
"short";
{
    "decimalFormat";
    {
        "1000-count-zero";
        "0 ألف",

```



```
        "1000-count-one";
    "0 ألف",
        "1000-count-two";
    "0 ألف",
        "1000-count-few";
    "0 آلاف",
        "1000-count-many";
    "0 ألف",
        "1000-count-other";
    "0 ألف",
        "10000-count-zero";
    "00 ألف",
        "10000-count-one";
    "00 ألف",
        "10000-count-two";
    "00 ألف",
        "10000-count-few";
    "00 ألف",
        "10000-count-many";
    "00 ألف",
        "10000-count-other";
    "00 ألف",
        "100000-count-zero";
    "000 ألف",
        "100000-count-one";
    "000 ألف",
        "100000-count-two";
    "000 ألف",
        "100000-count-few";
    "000 ألف",
        "100000-count-many";
    "000 ألف",
        "100000-count-other";
    "000 ألف",
        "1000000-count-zero";
    "0 مليون",
        "1000000-count-one";
    "0 مليون",
        "1000000-count-two";
    "0 مليون",
        "1000000-count-few";
    "0 مليون",
        "1000000-count-many";
    "0 مليون",
        "1000000-count-other";
    "0 مليون",
        "10000000-count-zero";
    "00 مليون",
        "10000000-count-one";
    "00 مليون",
        "10000000-count-two";
    "00 مليون",
        "10000000-count-few";
    "00 مليون",
        "10000000-count-many";
    "00 مليون",
        "10000000-count-other";
```

```

"00 مليو",
    "100000000-count-zero";
"000 مليو",
    "100000000-count-one";
"000 مليو",
    "100000000-count-two";
"000 مليو",
    "100000000-count-few";
"000 مليو",
    "100000000-count-many";
"000 مليو",
    "100000000-count-other";
"000 مليو",
    "1000000000-count-zero";
"0 مليا",
    "1000000000-count-one";
"0 مليا",
    "1000000000-count-two";
"0 مليا",
    "1000000000-count-few";
"0 مليا",
    "1000000000-count-many";
"0 مليا",
    "1000000000-count-other";
"0 مليا",
    "10000000000-count-zero";
"00 مليا",
    "10000000000-count-one";
"00 مليا",
    "10000000000-count-two";
"00 مليا",
    "10000000000-count-few";
"00 مليا",
    "10000000000-count-many";
"00 مليا",
    "10000000000-count-other";
"00 مليا",
    "100000000000-count-zero";
"000 مليا",
    "100000000000-count-one";
"000 مليا",
    "100000000000-count-two";
"000 مليا",
    "100000000000-count-few";
"000 مليا",
    "100000000000-count-many";
"000 مليا",
    "100000000000-count-other";
"000 مليا",
    "1000000000000-count-zero";
"0 ترليو",
    "1000000000000-count-one";
"0 ترليو",
    "1000000000000-count-two";
"0 ترليو",
    "1000000000000-count-few";
"0 ترليو",

```

```

        "1000000000000-count-many";
        "0 ترلیو",
        "1000000000000-count-other";
        "0 ترلیو",
        "1000000000000-count-zero";
        "00 ترلیو",
        "1000000000000-count-one";
        "00 ترلیو",
        "1000000000000-count-two";
        "00 ترلیو",
        "1000000000000-count-few";
        "00 ترلیو",
        "1000000000000-count-many";
        "00 ترلیو",
        "1000000000000-count-other";
        "00 ترلیو",
        "1000000000000-count-zero";
        "000 ترلیو",
        "1000000000000-count-one";
        "000 ترلیو",
        "1000000000000-count-two";
        "000 ترلیو",
        "1000000000000-count-few";
        "000 ترلیو",
        "1000000000000-count-many";
        "000 ترلیو",
        "1000000000000-count-other";
        "000 ترلیو";
    }
}
}
"scientificFormats-numberSystem-arab";
{
    "standard";
    "#E0";
}
"scientificFormats-numberSystem-latn";
{
    "standard";
    "#E0";
}
"percentFormats-numberSystem-arab";
{
    "standard";
    "#,##0 %";
}
"percentFormats-numberSystem-latn";
{
    "standard";
    "#,##0%";
}
"currencyFormats-numberSystem-arab";
{
    "currencySpacing";
    {
        "beforeCurrency";
        {

```

```

        "currencyMatch";
        "[:^S:]",
        "surroundingMatch";
        "[:digit:]",
        "insertBetween";
        " ";
    }
    "afterCurrency";
    {
        "currencyMatch";
        "[:^S:]",
        "surroundingMatch";
        "[:digit:]",
        "insertBetween";
        " ";
    }
}
"standard";
"#,##0.00 ¤",
    "accounting";
"#,##0.00 ¤",
    "unitPattern-count-zero";
"{0} {1}",
    "unitPattern-count-one";
"{0} {1}",
    "unitPattern-count-two";
"{0} {1}",
    "unitPattern-count-few";
"{0} {1}",
    "unitPattern-count-many";
"{0} {1}",
    "unitPattern-count-other";
"{0} {1}";
}
"currencyFormats-numberSystem-latn";
{
    "currencySpacing";
    {
        "beforeCurrency";
        {
            "currencyMatch";
            "[:^S:]",
            "surroundingMatch";
            "[:digit:]",
            "insertBetween";
            " ";
        }
        "afterCurrency";
        {
            "currencyMatch";
            "[:^S:]",
            "surroundingMatch";
            "[:digit:]",
            "insertBetween";
            " ";
        }
    }
}

```

```

"standard";
"¤ #,##0.00",
    "accounting";
"¤#,##0.00; (¤#,##0.00) ",
    "short";
{
    "standard";
    {
        "1000-count-zero";
        "¤ 0 ألف",
            "1000-count-one";
        "¤ 0 ألف",
            "1000-count-two";
        "¤ 0 ألف",
            "1000-count-few";
        "¤ 0 ألف",
            "1000-count-many";
        "¤ 0 ألف",
            "1000-count-other";
        "¤ 0 ألف",
            "10000-count-zero";
        "¤ 00 ألف",
            "10000-count-one";
        "¤ 00 ألف",
            "10000-count-two";
        "¤ 00 ألف",
            "10000-count-few";
        "¤ 00 ألف",
            "10000-count-many";
        "¤ 00 ألف",
            "10000-count-other";
        "¤ 00 ألف",
            "100000-count-zero";
        "¤ 000 ألف",
            "100000-count-one";
        "¤ 000 ألف",
            "100000-count-two";
        "¤ 000 ألف",
            "100000-count-few";
        "¤ 000 ألف",
            "100000-count-many";
        "¤ 000 ألف",
            "100000-count-other";
        "¤ 000 ألف",
            "1000000-count-zero";
        "¤ 0 مليون",
            "1000000-count-one";
        "¤ 0 مليون",
            "1000000-count-two";
        "¤ 0 مليون",
            "1000000-count-few";
        "¤ 0 مليون",
            "1000000-count-many";
        "¤ 0 مليون",
            "1000000-count-other";
        "¤ 0 مليون",
            "10000000-count-zero";
    }
}

```

```

"٠ ٠٠ مليون",
  "10000000-count-one";
"٠ ٠٠ مليون",
  "10000000-count-two";
"٠ ٠٠ مليون",
  "10000000-count-few";
"٠ ٠٠ مليون",
  "10000000-count-many";
"٠ ٠٠ مليون",
  "10000000-count-other";
"٠ ٠٠ مليون",
  "100000000-count-zero";
"٠ ٠٠٠ مليون",
  "100000000-count-one";
"٠ ٠٠٠ مليون",
  "100000000-count-two";
"٠ ٠٠٠ مليون",
  "100000000-count-few";
"٠ ٠٠٠ مليون",
  "100000000-count-many";
"٠ ٠٠٠ مليون",
  "100000000-count-other";
"٠ ٠٠٠ مليون",
  "1000000000-count-zero";
"٠ ٠ مليا",
  "1000000000-count-one";
"٠ ٠ مليا",
  "1000000000-count-two";
"٠ ٠ مليا",
  "1000000000-count-few";
"٠ ٠ مليا",
  "1000000000-count-many";
"٠ ٠ مليا",
  "1000000000-count-other";
"٠ ٠ مليا",
  "10000000000-count-zero";
"٠ ٠٠ مليا",
  "10000000000-count-one";
"٠ ٠٠ مليا",
  "10000000000-count-two";
"٠ ٠٠ مليا",
  "10000000000-count-few";
"٠ ٠٠ مليا",
  "10000000000-count-many";
"٠ ٠٠ مليا",
  "10000000000-count-other";
"٠ ٠٠ مليا",
  "100000000000-count-zero";
"٠ ٠٠٠ مليا",
  "100000000000-count-one";
"٠ ٠٠٠ مليا",
  "100000000000-count-two";
"٠ ٠٠٠ مليا",
  "100000000000-count-few";
"٠ ٠٠٠ مليا",
  "100000000000-count-many";
"٠ ٠٠٠ مليا",

```

```

        "100000000000-count-other";
        "¤ 000 مليا",
        "100000000000-count-zero";
        "¤ 0 ترليو",
        "100000000000-count-one";
        "¤ 0 ترليو",
        "100000000000-count-two";
        "¤ 0 ترليو",
        "100000000000-count-few";
        "¤ 0 ترليو",
        "100000000000-count-many";
        "¤ 0 ترليو",
        "100000000000-count-other";
        "¤ 0 ترليو",
        "100000000000-count-zero";
        "¤ 00 ترليو",
        "100000000000-count-one";
        "¤ 00 ترليو",
        "100000000000-count-two";
        "¤ 00 ترليو",
        "100000000000-count-few";
        "¤ 00 ترليو",
        "100000000000-count-many";
        "¤ 00 ترليو",
        "100000000000-count-other";
        "¤ 00 ترليو",
        "100000000000-count-zero";
        "¤ 000 ترليو",
        "100000000000-count-one";
        "¤ 000 ترليو",
        "100000000000-count-two";
        "¤ 000 ترليو",
        "100000000000-count-few";
        "¤ 000 ترليو",
        "100000000000-count-many";
        "¤ 000 ترليو",
        "100000000000-count-other";
        "¤ 000 ترليو";
    }
}
"unitPattern-count-zero";
"{0} {1}",
    "unitPattern-count-one";
"{0} {1}",
    "unitPattern-count-two";
"{0} {1}",
    "unitPattern-count-few";
"{0} {1}",
    "unitPattern-count-many";
"{0} {1}",
    "unitPattern-count-other";
"{0} {1}";
}
"miscPatterns-numberSystem-arab";
{
    "atLeast";
    "+{0}",

```

```

        "range";
        "{0}-{1}";
    }
    "miscPatterns-numberSystem-latn";
    {
        "atLeast";
        "+{0}",
        "range";
        "{0}-{1}";
    }
    }
}

```

TIMEZONENAMES.JSON

```

{
  "main": {
    "ar": {
      "identity": {
        "version": {
          "_number": "$Revision: 12879 $",
          "_cldrVersion": "30.0.3"
        },
        "language": "ar"
      },
      "dates": {
        "timeZoneNames": {
          "hourFormat": "+HH:mm;-HH:mm",
          "gmtFormat": "0{جرينتش}",
          "gmtZeroFormat": "جرينتش",
          "regionFormat": "0{توقيت}",
          "regionFormat-type-daylight": "الصيفي 0{توقيت}",
          "regionFormat-type-standard": "الرسمي 0{توقيت}",
          "fallbackFormat": "{1} ({0})",
          "zone": {
            "America": {
              "Adak": {
                "exemplarCity": "أداك"
              },
              "Anchorage": {
                "exemplarCity": "أنشوراج"
              },
              "Anguilla": {
                "exemplarCity": "أنغيلا"
              },
              "Antigua": {
                "exemplarCity": "أنتيغوا"
              },
              "Araguaina": {
                "exemplarCity": "أروجوانيا"
              },
              "Argentina": {
                "Rio_Gallegos": {
                  "exemplarCity": "ريو جالييوس"
                }
              }
            }
          }
        }
      }
    }
  }
}

```



```

    },
    "San_Juan": {
      "exemplarCity": "سان خوان"
    },
    "Ushuaia": {
      "exemplarCity": "أشوا"
    },
    "La_Rioja": {
      "exemplarCity": "لا ريوجا"
    },
    "San_Luis": {
      "exemplarCity": "سان لويس"
    },
    "Salta": {
      "exemplarCity": "سالطا"
    },
    "Tucuman": {
      "exemplarCity": "تاكمان"
    }
  },
  "Aruba": {
    "exemplarCity": "أروبا"
  },
  "Asuncion": {
    "exemplarCity": "أسونسيون"
  },
  "Bahia": {
    "exemplarCity": "باهيا"
  },
  "Bahia_Banderas": {
    "exemplarCity": "باهيا بانديراس"
  },
  "Barbados": {
    "exemplarCity": "بربادوس"
  },
  "Belem": {
    "exemplarCity": "بلم"
  },
  "Belize": {
    "exemplarCity": "بليز"
  },
  "Blanc-Sablon": {
    "exemplarCity": "بلانك-سابلون"
  },
  "Boa_Vista": {
    "exemplarCity": "باو فيستا"
  },
  "Bogota": {
    "exemplarCity": "بوغوتا"
  },
  "Boise": {
    "exemplarCity": "بويس"
  },
  "Buenos_Aires": {
    "exemplarCity": "بوينوس أيرس"
  },
  "Cambridge_Bay": {

```

```

    "exemplarCity": "كامبرديج باي"
  },
  "Campo_Grande": {
    "exemplarCity": "كومبو جراند"
  },
  "Cancun": {
    "exemplarCity": "كانكون"
  },
  "Caracas": {
    "exemplarCity": "كاراكاس"
  },
  "Catamarca": {
    "exemplarCity": "كاتاماركا"
  },
  "Cayenne": {
    "exemplarCity": "كايين"
  },
  "Cayman": {
    "exemplarCity": "كيمان"
  },
  "Chicago": {
    "exemplarCity": "شيكاغو"
  },
  "Chihuahua": {
    "exemplarCity": "تشيوواوا"
  },
  "Coral_Harbour": {
    "exemplarCity": "كورانل هاربر"
  },
  "Cordoba": {
    "exemplarCity": "كوردوبا"
  },
  "Costa_Rica": {
    "exemplarCity": "كوستاريكا"
  },
  "Creston": {
    "exemplarCity": "كريستون"
  },
  "Cuiaba": {
    "exemplarCity": "كيابا"
  },
  "Curacao": {
    "exemplarCity": "كوراكاو"
  },
  "Danmarkshavn": {
    "exemplarCity": "دانمرك شافن"
  },
  "Dawson": {
    "exemplarCity": "داوسان"
  },
  "Dawson_Creek": {
    "exemplarCity": "داوسن كريك"
  },
  "Denver": {
    "exemplarCity": "دنفر"
  },
  "Detroit": {

```

```

    "exemplarCity": "ديترويت"
  },
  "Dominica": {
    "exemplarCity": "دومينيكا"
  },
  "Edmonton": {
    "exemplarCity": "ايدمونتون"
  },
  "Eirunepe": {
    "exemplarCity": "ايرونبي"
  },
  "El_Salvador": {
    "exemplarCity": "السلفادور"
  },
  "Fort_Nelson": {
    "exemplarCity": "فورت نيلسون"
  },
  "Fortaleza": {
    "exemplarCity": "فورتاليزا"
  },
  "Glace_Bay": {
    "exemplarCity": "جلاس باي"
  },
  "Godthab": {
    "exemplarCity": "غودثاب"
  },
  "Goose_Bay": {
    "exemplarCity": "جوس باي"
  },
  "Grand_Turk": {
    "exemplarCity": "غراند ترك"
  },
  "Grenada": {
    "exemplarCity": "جرينادا"
  },
  "Guadeloupe": {
    "exemplarCity": "غوادلوب"
  },
  "Guatemala": {
    "exemplarCity": "غواتيمالا"
  },
  "Guayaquil": {
    "exemplarCity": "غواياكويل"
  },
  "Guyana": {
    "exemplarCity": "غيانا"
  },
  "Halifax": {
    "exemplarCity": "هاليفاكس"
  },
  "Havana": {
    "exemplarCity": "هافانا"
  },
  "Hermosillo": {
    "exemplarCity": "هيرموسيلو"
  },
  "Indiana": {

```

```

    "Vincennes": {
      "exemplarCity": "فينسينس"
    },
    "Petersburg": {
      "exemplarCity": "بيترسبرغ"
    },
    "Tell_City": {
      "exemplarCity": "مدينة تل، إنديانا"
    },
    "Knox": {
      "exemplarCity": "كونكس"
    },
    "Winamac": {
      "exemplarCity": "ويناماك"
    },
    "Marengo": {
      "exemplarCity": "مارنجو"
    },
    "Vevay": {
      "exemplarCity": "فيفاي"
    }
  },
  "Indianapolis": {
    "exemplarCity": "إنديانا بوليس"
  },
  "Inuvik": {
    "exemplarCity": "اينوفيك"
  },
  "Iqaluit": {
    "exemplarCity": "اكويلت"
  },
  "Jamaica": {
    "exemplarCity": "جامايكا"
  },
  "Jujuy": {
    "exemplarCity": "جوجو"
  },
  "Juneau": {
    "exemplarCity": "جوناي"
  },
  "Kentucky": {
    "Monticello": {
      "exemplarCity": "مونتي سيلو"
    }
  },
  "Kralendijk": {
    "exemplarCity": "كرالنديك"
  },
  "La_Paz": {
    "exemplarCity": "لا باز"
  },
  "Lima": {
    "exemplarCity": "ليما"
  },
  "Los_Angeles": {
    "exemplarCity": "لوس انجلوس"
  },
}

```

```

"Louisville": {
  "exemplarCity": "لويس فيل"
},
"Lower_Princes": {
  "exemplarCity": "حي الأمير السفلي"
},
"Maceio": {
  "exemplarCity": "ماشيو"
},
"Managua": {
  "exemplarCity": "ماناغوا"
},
"Manaus": {
  "exemplarCity": "ماناوس"
},
"Marigot": {
  "exemplarCity": "ماريغوت"
},
"Martinique": {
  "exemplarCity": "المارتينيك"
},
"Matamoros": {
  "exemplarCity": "ماتاموروس"
},
"Mazatlan": {
  "exemplarCity": "مازاتلان"
},
"Mendoza": {
  "exemplarCity": "ميندوزا"
},
"Menominee": {
  "exemplarCity": "مينوميني"
},
"Merida": {
  "exemplarCity": "ميريدا"
},
"Metlakatla": {
  "exemplarCity": "ميتلاكاتلا"
},
"Mexico_City": {
  "exemplarCity": "مدينة المكسيك"
},
"Miquelon": {
  "exemplarCity": "ميكولن"
},
"Moncton": {
  "exemplarCity": "وينكتون"
},
"Monterrey": {
  "exemplarCity": "مونتيري"
},
"Montevideo": {
  "exemplarCity": "مونتيفيديو"
},
"Montserrat": {
  "exemplarCity": "مونتسيرات"
},

```

```

"Nassau": {
  "exemplarCity": "ناسو"
},
"New_York": {
  "exemplarCity": "نيويورك"
},
"Nipigon": {
  "exemplarCity": "نيبيجون"
},
"Nome": {
  "exemplarCity": "نوم"
},
"Noronha": {
  "exemplarCity": "نوروناه"
},
"North_Dakota": {
  "Beulah": {
    "exemplarCity": "بيولا، داکوتا الشمالية"
  },
  "New_Salem": {
    "exemplarCity": "نيو ساليم"
  },
  "Center": {
    "exemplarCity": "سنتر"
  }
},
"Ojinaga": {
  "exemplarCity": "أوجيناغا"
},
"Panama": {
  "exemplarCity": "بنما"
},
"Pangnirtung": {
  "exemplarCity": "بانجينتينج"
},
"Paramaribo": {
  "exemplarCity": "باراماريبو"
},
"Phoenix": {
  "exemplarCity": "فينكس"
},
"Port-au-Prince": {
  "exemplarCity": "بورت أو برنس"
},
"Port_of_Spain": {
  "exemplarCity": "بورت أوف سبين"
},
"Porto_Velho": {
  "exemplarCity": "بورتو فيلو"
},
"Puerto_Rico": {
  "exemplarCity": "بورتوريكو"
},
"Rainy_River": {
  "exemplarCity": "راني ريفر"
},
"Rankin_Inlet": {

```

```

    "exemplarCity": "رانكن انلت"
  },
  "Recife": {
    "exemplarCity": "ريسيف"
  },
  "Regina": {
    "exemplarCity": "ريجينا"
  },
  "Resolute": {
    "exemplarCity": "ريزولوت"
  },
  "Rio_Branco": {
    "exemplarCity": "ريوبرانكو"
  },
  "Santa_Isabel": {
    "exemplarCity": "سانتا ايزابيل"
  },
  "Santarem": {
    "exemplarCity": "سانتاريم"
  },
  "Santiago": {
    "exemplarCity": "سانتياغو"
  },
  "Santo_Domingo": {
    "exemplarCity": "سانتو دومينغو"
  },
  "Sao_Paulo": {
    "exemplarCity": "ساو باولو"
  },
  "Scoresbysund": {
    "exemplarCity": "سكورسبيسند"
  },
  "Sitka": {
    "exemplarCity": "سيتكا"
  },
  "St_Barthelemy": {
    "exemplarCity": "سانت بارتيليمي"
  },
  "St_Johns": {
    "exemplarCity": "سانت جونز"
  },
  "St_Kitts": {
    "exemplarCity": "سانت كيتس"
  },
  "St_Lucia": {
    "exemplarCity": "سانت لوشيا"
  },
  "St_Thomas": {
    "exemplarCity": "سانت توماس"
  },
  "St_Vincent": {
    "exemplarCity": "سانت فنسنت"
  },
  "Swift_Current": {
    "exemplarCity": "سوفت كارنت"
  },
  "Tegucigalpa": {

```

```

        "exemplarCity": "تيغوسيغالبا"
    },
    "Thule": {
        "exemplarCity": "ثيل"
    },
    "Thunder_Bay": {
        "exemplarCity": "ثندر باي"
    },
    "Tijuana": {
        "exemplarCity": "تيخوانا"
    },
    "Toronto": {
        "exemplarCity": "تورونتو"
    },
    "Tortola": {
        "exemplarCity": "تورتولا"
    },
    "Vancouver": {
        "exemplarCity": "فانكوفر"
    },
    "Whitehorse": {
        "exemplarCity": "وايت هورس"
    },
    "Winnipeg": {
        "exemplarCity": "وينيبيج"
    },
    "Yakutat": {
        "exemplarCity": "ياكوتات"
    },
    "Yellowknife": {
        "exemplarCity": "يلونيف"
    }
},
"Atlantic": {
    "Azores": {
        "exemplarCity": "أزورس"
    },
    "Bermuda": {
        "exemplarCity": "برمودا"
    },
    "Canary": {
        "exemplarCity": "كناري"
    },
    "Cape_Verde": {
        "exemplarCity": "الرأس الأخضر"
    },
    "Faeroe": {
        "exemplarCity": "فارو"
    },
    "Madeira": {
        "exemplarCity": "ماديرا"
    },
    "Reykjavik": {
        "exemplarCity": "ريكيافيك"
    },
    "South_Georgia": {
        "exemplarCity": "جورجيا الجنوبية"
    }
}

```



```

    },
    "St_Helena": {
      "exemplarCity": "سانت هيلينا"
    },
    "Stanley": {
      "exemplarCity": "استانلي"
    }
  },
  "Europe": {
    "Amsterdam": {
      "exemplarCity": "أمستردام"
    },
    "Andorra": {
      "exemplarCity": "أندورا"
    },
    "Astrakhan": {
      "exemplarCity": "أستراخان"
    },
    "Athens": {
      "exemplarCity": "أثينا"
    },
    "Belgrade": {
      "exemplarCity": "بلغراد"
    },
    "Berlin": {
      "exemplarCity": "برلين"
    },
    "Bratislava": {
      "exemplarCity": "براتيسلافا"
    },
    "Brussels": {
      "exemplarCity": "بروكسل"
    },
    "Bucharest": {
      "exemplarCity": "بوخارست"
    },
    "Budapest": {
      "exemplarCity": "بودابست"
    },
    "Busingen": {
      "exemplarCity": "بوسنغن"
    },
    "Chisinau": {
      "exemplarCity": "تشيسيناو"
    },
    "Copenhagen": {
      "exemplarCity": "كوبنهاغن"
    },
    "Dublin": {
      "long": {
        "daylight": "توقيت أيرلندا الرسمي"
      },
      "exemplarCity": "دبلن"
    },
    "Gibraltar": {
      "exemplarCity": "جبل طارق"
    }
  },

```

```
"Guernsey": {
  "exemplarCity": "غيرنسي"
},
"Helsinki": {
  "exemplarCity": "هلسنكي"
},
"Isle_of_Man": {
  "exemplarCity": "جزيرة مان"
},
"Istanbul": {
  "exemplarCity": "إسطنبول"
},
"Jersey": {
  "exemplarCity": "جيرسي"
},
"Kaliningrad": {
  "exemplarCity": "كالينجراد"
},
"Kiev": {
  "exemplarCity": "كييف"
},
"Kirov": {
  "exemplarCity": "كيروف"
},
"Lisbon": {
  "exemplarCity": "لشبونة"
},
"Ljubljana": {
  "exemplarCity": "ليوبليانا"
},
"London": {
  "long": {
    "daylight": "توقيت بريطانيا الصيفي"
  },
  "exemplarCity": "لندن"
},
"Luxembourg": {
  "exemplarCity": "لوكسمبورغ"
},
"Madrid": {
  "exemplarCity": "مدريد"
},
"Malta": {
  "exemplarCity": "مالطة"
},
"Mariehamn": {
  "exemplarCity": "ماريهامن"
},
"Minsk": {
  "exemplarCity": "مينسك"
},
"Monaco": {
  "exemplarCity": "موناكو"
},
"Moscow": {
  "exemplarCity": "موسكو"
},
},
```

```
"Oslo": {
  "exemplarCity": "أوسلو"
},
"Paris": {
  "exemplarCity": "باريس"
},
"Podgorica": {
  "exemplarCity": "بودغوريكا"
},
"Prague": {
  "exemplarCity": "براغ"
},
"Riga": {
  "exemplarCity": "ريغا"
},
"Rome": {
  "exemplarCity": "روما"
},
"Samara": {
  "exemplarCity": "سمراء"
},
"San_Marino": {
  "exemplarCity": "سان مارينو"
},
"Sarajevo": {
  "exemplarCity": "سراييفو"
},
"Simferopol": {
  "exemplarCity": "سيمفروبول"
},
"Skopje": {
  "exemplarCity": "سكوبي"
},
"Sofia": {
  "exemplarCity": "صوفيا"
},
"Stockholm": {
  "exemplarCity": "ستوكهولم"
},
"Tallinn": {
  "exemplarCity": "تالين"
},
"Tirane": {
  "exemplarCity": "تيرانا"
},
"Ulyanovsk": {
  "exemplarCity": "أوليانوفسك"
},
"Uzhgorod": {
  "exemplarCity": "أوزجروود"
},
"Vaduz": {
  "exemplarCity": "فادوز"
},
"Vatican": {
  "exemplarCity": "الفاتيكان"
},
},
```

```

"Vienna": {
  "exemplarCity": "فيينا"
},
"Vilnius": {
  "exemplarCity": "فيلنيوس"
},
"Volgograd": {
  "exemplarCity": "فولوجراد"
},
"Warsaw": {
  "exemplarCity": "وارسو"
},
"Zagreb": {
  "exemplarCity": "زغرب"
},
"Zaporozhye": {
  "exemplarCity": "زابوروزي"
},
"Zurich": {
  "exemplarCity": "زيورخ"
}
},
"Africa": {
  "Abidjan": {
    "exemplarCity": "أبيدجان"
  },
  "Accra": {
    "exemplarCity": "أكرا"
  },
  "Addis_Ababa": {
    "exemplarCity": "أديس أبابا"
  },
  "Algiers": {
    "exemplarCity": "الجزائر"
  },
  "Asmera": {
    "exemplarCity": "أسمره"
  },
  "Bamako": {
    "exemplarCity": "باماكو"
  },
  "Bangui": {
    "exemplarCity": "بانغوي"
  },
  "Banjul": {
    "exemplarCity": "بانجول"
  },
  "Bissau": {
    "exemplarCity": "بيساو"
  },
  "Blantyre": {
    "exemplarCity": "بلانتيير"
  },
  "Brazzaville": {
    "exemplarCity": "برازافيل"
  },
  "Bujumbura": {

```

```
    "exemplarCity": "بوجومبورا"
  },
  "Cairo": {
    "exemplarCity": "القاهرة"
  },
  "Casablanca": {
    "exemplarCity": "الدار البيضاء"
  },
  "Ceuta": {
    "exemplarCity": "سيتا"
  },
  "Conakry": {
    "exemplarCity": "كوناكري"
  },
  "Dakar": {
    "exemplarCity": "داكار"
  },
  "Dar_es_Salaam": {
    "exemplarCity": "دار السلام"
  },
  "Djibouti": {
    "exemplarCity": "جيبوتي"
  },
  "Douala": {
    "exemplarCity": "دوالا"
  },
  "El_Aaiun": {
    "exemplarCity": "العيون"
  },
  "Freetown": {
    "exemplarCity": "فري تاون"
  },
  "Gaborone": {
    "exemplarCity": "غابورون"
  },
  "Harare": {
    "exemplarCity": "هراري"
  },
  "Johannesburg": {
    "exemplarCity": "جوهانسبرغ"
  },
  "Juba": {
    "exemplarCity": "جوبا"
  },
  "Kampala": {
    "exemplarCity": "كامبالا"
  },
  "Khartoum": {
    "exemplarCity": "الخرطوم"
  },
  "Kigali": {
    "exemplarCity": "كيغالي"
  },
  "Kinshasa": {
    "exemplarCity": "كينشاسا"
  },
  "Lagos": {
```

```

    "exemplarCity": "لاغوس"
  },
  "Libreville": {
    "exemplarCity": "ليبرفيل"
  },
  "Lome": {
    "exemplarCity": "لومي"
  },
  "Luanda": {
    "exemplarCity": "لواندا"
  },
  "Lubumbashi": {
    "exemplarCity": "لومبباشا"
  },
  "Lusaka": {
    "exemplarCity": "لوساكا"
  },
  "Malabo": {
    "exemplarCity": "مالابو"
  },
  "Maputo": {
    "exemplarCity": "مابوتو"
  },
  "Maseru": {
    "exemplarCity": "ماسيرو"
  },
  "Mbabane": {
    "exemplarCity": "مباباني"
  },
  "Mogadishu": {
    "exemplarCity": "مقديشيو"
  },
  "Monrovia": {
    "exemplarCity": "مونروفيا"
  },
  "Nairobi": {
    "exemplarCity": "نيروبي"
  },
  "Ndjamena": {
    "exemplarCity": "نجامينا"
  },
  "Niamey": {
    "exemplarCity": "نيامي"
  },
  "Nouakchott": {
    "exemplarCity": "نواكشوط"
  },
  "Ouagadougou": {
    "exemplarCity": "واغادوغو"
  },
  "Porto-Novo": {
    "exemplarCity": "بورتو نوفو"
  },
  "Sao_Tome": {
    "exemplarCity": "ساو تومي"
  },
  "Tripoli": {

```

```

        "exemplarCity": "طرابلس"
    },
    "Tunis": {
        "exemplarCity": "تونس"
    },
    "Windhoek": {
        "exemplarCity": "ويندهوك"
    }
},
"Asia": {
    "Aden": {
        "exemplarCity": "عدن"
    },
    "Almaty": {
        "exemplarCity": "ألماتي"
    },
    "Amman": {
        "exemplarCity": "عمان"
    },
    "Anadyr": {
        "exemplarCity": "أندير"
    },
    "Aqtau": {
        "exemplarCity": "أكتاو"
    },
    "Aqtobe": {
        "exemplarCity": "أكتوب"
    },
    "Ashgabat": {
        "exemplarCity": "عشق آباد"
    },
    "Baghdad": {
        "exemplarCity": "بغداد"
    },
    "Bahrain": {
        "exemplarCity": "البحرين"
    },
    "Baku": {
        "exemplarCity": "باكو"
    },
    "Bangkok": {
        "exemplarCity": "بانكوك"
    },
    "Barnaul": {
        "exemplarCity": "بارناول"
    },
    "Beirut": {
        "exemplarCity": "بيروت"
    },
    "Bishkek": {
        "exemplarCity": "بشكيك"
    },
    "Brunei": {
        "exemplarCity": "بروناي"
    },
    "Calcutta": {
        "exemplarCity": "كالكتا"
    }
}

```

```
    },
    "Chita": {
      "exemplarCity": "تشيتا"
    },
    "Choibalsan": {
      "exemplarCity": "تشوبالسان"
    },
    "Colombo": {
      "exemplarCity": "كولومبو"
    },
    "Damascus": {
      "exemplarCity": "دمشق"
    },
    "Dhaka": {
      "exemplarCity": "دكا"
    },
    "Dili": {
      "exemplarCity": "ديلي"
    },
    "Dubai": {
      "exemplarCity": "دبي"
    },
    "Dushanbe": {
      "exemplarCity": "دوشانبي"
    },
    "Gaza": {
      "exemplarCity": "غزة"
    },
    "Hebron": {
      "exemplarCity": "هيبرون (مدينة الخليل)"
    },
    "Hong_Kong": {
      "exemplarCity": "هونغ كونغ"
    },
    "Hovd": {
      "exemplarCity": "هوفد"
    },
    "Irkutsk": {
      "exemplarCity": "ايركيتسك"
    },
    "Jakarta": {
      "exemplarCity": "جاكرتا"
    },
    "Jayapura": {
      "exemplarCity": "جاياپورا"
    },
    "Jerusalem": {
      "exemplarCity": "القدس"
    },
    "Kabul": {
      "exemplarCity": "كابول"
    },
    "Kamchatka": {
      "exemplarCity": "كامتشاتكا"
    },
    "Karachi": {
      "exemplarCity": "كراتشي"
```



```

    },
    "Katmandu": {
      "exemplarCity": "كاتماندو"
    },
    "Khandyga": {
      "exemplarCity": "خانديجا"
    },
    "Krasnoyarsk": {
      "exemplarCity": "كراسنويارسك"
    },
    "Kuala_Lumpur": {
      "exemplarCity": "كوالا لامبور"
    },
    "Kuching": {
      "exemplarCity": "كيشينج"
    },
    "Kuwait": {
      "exemplarCity": "الكويت"
    },
    "Macau": {
      "exemplarCity": "ماكاو"
    },
    "Magadan": {
      "exemplarCity": "مجادن"
    },
    "Makassar": {
      "exemplarCity": "ماكسار"
    },
    "Manila": {
      "exemplarCity": "مانيلا"
    },
    "Muscat": {
      "exemplarCity": "مسقط"
    },
    "Nicosia": {
      "exemplarCity": "نيقوسيا"
    },
    "Novokuznetsk": {
      "exemplarCity": "نوفوكوزنتسك"
    },
    "Novosibirsk": {
      "exemplarCity": "نوفوسبيرسك"
    },
    "Omsk": {
      "exemplarCity": "أومسك"
    },
    "Oral": {
      "exemplarCity": "أورال"
    },
    "Phnom_Penh": {
      "exemplarCity": "بنوم بنه"
    },
    "Pontianak": {
      "exemplarCity": "بونتيانك"
    },
    "Pyongyang": {
      "exemplarCity": "بيونغ يانغ"
    }
  },

```

```
},
"Qatar": {
  "exemplarCity": "قطر"
},
"Qyzylorda": {
  "exemplarCity": "كيزيلوردا"
},
"Rangoon": {
  "exemplarCity": "رانغون"
},
"Riyadh": {
  "exemplarCity": "الرياض"
},
"Saigon": {
  "exemplarCity": "مدينة هو تشي منه"
},
"Sakhalin": {
  "exemplarCity": "سكالين"
},
"Samarkand": {
  "exemplarCity": "سمرقند"
},
"Seoul": {
  "exemplarCity": "سول"
},
"Shanghai": {
  "exemplarCity": "شنغهاي"
},
"Singapore": {
  "exemplarCity": "سنغافورة"
},
"Srednekolymsk": {
  "exemplarCity": "سريدنكوليمسك"
},
"Taipei": {
  "exemplarCity": "تايبه"
},
"Tashkent": {
  "exemplarCity": "تشقند"
},
"Tbilisi": {
  "exemplarCity": "تبليسي"
},
"Tehran": {
  "exemplarCity": "تهران"
},
"Thimphu": {
  "exemplarCity": "تيمفو"
},
"Tokyo": {
  "exemplarCity": "طوكيو"
},
"Tomsk": {
  "exemplarCity": "تومسك"
},
"Ulaanbaatar": {
  "exemplarCity": "آلانباتار"
}
```

```

    },
    "Urumqi": {
      "exemplarCity": "أرومكي"
    },
    "Ust-Nera": {
      "exemplarCity": "أوست نيرا"
    },
    "Vientiane": {
      "exemplarCity": "فيانتيان"
    },
    "Vladivostok": {
      "exemplarCity": "فلاديفوستك"
    },
    "Yakutsk": {
      "exemplarCity": "ياكتسك"
    },
    "Yekaterinburg": {
      "exemplarCity": "يكاترنبيرج"
    },
    "Yerevan": {
      "exemplarCity": "يريفان"
    }
  },
  "Indian": {
    "Antananarivo": {
      "exemplarCity": "أنتاناناريفو"
    },
    "Chagos": {
      "exemplarCity": "تشاغوس"
    },
    "Christmas": {
      "exemplarCity": "كريسماس"
    },
    "Cocos": {
      "exemplarCity": "كوكوس"
    },
    "Comoro": {
      "exemplarCity": "جزر القمر"
    },
    "Kerguelen": {
      "exemplarCity": "كيرغويلين"
    },
    "Mahe": {
      "exemplarCity": "ماهي"
    },
    "Maldives": {
      "exemplarCity": "المالديف"
    },
    "Mauritius": {
      "exemplarCity": "موريشيوس"
    },
    "Mayotte": {
      "exemplarCity": "مايوت"
    },
    "Reunion": {
      "exemplarCity": "ريونيون"
    }
  }
}

```

```

    },
    "Australia": {
      "Adelaide": {
        "exemplarCity": "أديليد"
      },
      "Brisbane": {
        "exemplarCity": "برسبان"
      },
      "Broken_Hill": {
        "exemplarCity": "بروكن هيل"
      },
      "Currie": {
        "exemplarCity": "كوري"
      },
      "Darwin": {
        "exemplarCity": "دارون"
      },
      "Eucla": {
        "exemplarCity": "أوكلأ"
      },
      "Hobart": {
        "exemplarCity": "هوبارت"
      },
      "Lindeman": {
        "exemplarCity": "ليندمان"
      },
      "Lord_Howe": {
        "exemplarCity": "لورد هاو"
      },
      "Melbourne": {
        "exemplarCity": "ميلبورن"
      },
      "Perth": {
        "exemplarCity": "برثا"
      },
      "Sydney": {
        "exemplarCity": "سيدني"
      }
    },
    "Pacific": {
      "Apia": {
        "exemplarCity": "أبيا"
      },
      "Auckland": {
        "exemplarCity": "أوكلاند"
      },
      "Bougainville": {
        "exemplarCity": "بوغانفيل"
      },
      "Chatham": {
        "exemplarCity": "تشاثام"
      },
      "Easter": {
        "exemplarCity": "استر"
      },
      "Efate": {
        "exemplarCity": "إيفات"
      }
    }
  }

```

```

    },
    "Enderbury": {
      "exemplarCity": "اندربيرج"
    },
    "Fakaofu": {
      "exemplarCity": "فاكاوفو"
    },
    "Fiji": {
      "exemplarCity": "فيجي"
    },
    "Funafuti": {
      "exemplarCity": "فونافوتي"
    },
    "Galapagos": {
      "exemplarCity": "جلاباجوس"
    },
    "Gambier": {
      "exemplarCity": "جامبير"
    },
    "Guadalcanal": {
      "exemplarCity": "غواد الكانال"
    },
    "Guam": {
      "exemplarCity": "غوام"
    },
    "Honolulu": {
      "exemplarCity": "هونولولو"
    },
    "Johnston": {
      "exemplarCity": "جونستون"
    },
    "Kiritimati": {
      "exemplarCity": "كيريتي ماتي"
    },
    "Kosrae": {
      "exemplarCity": "كوسرا"
    },
    "Kwajalein": {
      "exemplarCity": "كواجالين"
    },
    "Majuro": {
      "exemplarCity": "ماجورو"
    },
    "Marquesas": {
      "exemplarCity": "ماركيساس"
    },
    "Midway": {
      "exemplarCity": "ميدواي"
    },
    "Nauru": {
      "exemplarCity": "ناورو"
    },
    "Niue": {
      "exemplarCity": "نيوي"
    },
    "Norfolk": {
      "exemplarCity": "نورفولك"
    }
  },

```

```

    },
    "Noumea": {
      "exemplarCity": "نوميا"
    },
    "Pago_Pago": {
      "exemplarCity": "باغو باغو"
    },
    "Palau": {
      "exemplarCity": "بالاو"
    },
    "Pitcairn": {
      "exemplarCity": "بيتكيرن"
    },
    "Ponape": {
      "exemplarCity": "باناب"
    },
    "Port_Moresby": {
      "exemplarCity": "بور مورسبي"
    },
    "Rarotonga": {
      "exemplarCity": "راروتونغا"
    },
    "Saipan": {
      "exemplarCity": "سايبان"
    },
    "Tahiti": {
      "exemplarCity": "تاهيتي"
    },
    "Tarawa": {
      "exemplarCity": "تاراوا"
    },
    "Tongatapu": {
      "exemplarCity": "تونغاتابو"
    },
    "Truk": {
      "exemplarCity": "ترك"
    },
    "Wake": {
      "exemplarCity": "واك"
    },
    "Wallis": {
      "exemplarCity": "واليس"
    }
  },
  "Arctic": {
    "Longyearbyen": {
      "exemplarCity": "لونغجيربين"
    }
  },
  "Antarctica": {
    "Casey": {
      "exemplarCity": "كاساي"
    },
    "Davis": {
      "exemplarCity": "دافيز"
    },
    "DumontDUrville": {

```

```

    "exemplarCity": "دي مونت دو روفيل"
  },
  "Macquarie": {
    "exemplarCity": "ماكواري"
  },
  "Mawson": {
    "exemplarCity": "ماوسون"
  },
  "McMurdo": {
    "exemplarCity": "ماك موردو"
  },
  "Palmer": {
    "exemplarCity": "بالمير"
  },
  "Rothera": {
    "exemplarCity": "روثيرا"
  },
  "Syowa": {
    "exemplarCity": "سايووا"
  },
  "Troll": {
    "exemplarCity": "ترول"
  },
  "Vostok": {
    "exemplarCity": "فوستوك"
  }
},
"Etc": {
  "GMT": {
    "exemplarCity": "GMT"
  },
  "GMT1": {
    "exemplarCity": "GMT+1"
  },
  "GMT10": {
    "exemplarCity": "GMT+10"
  },
  "GMT11": {
    "exemplarCity": "GMT+11"
  },
  "GMT12": {
    "exemplarCity": "GMT+12"
  },
  "GMT2": {
    "exemplarCity": "GMT+2"
  },
  "GMT3": {
    "exemplarCity": "GMT+3"
  },
  "GMT4": {
    "exemplarCity": "GMT+4"
  },
  "GMT5": {
    "exemplarCity": "GMT+5"
  },
  "GMT6": {
    "exemplarCity": "GMT+6"
  }
}

```

```

    },
    "GMT7": {
      "exemplarCity": "GMT+7"
    },
    "GMT8": {
      "exemplarCity": "GMT+8"
    },
    "GMT9": {
      "exemplarCity": "GMT+9"
    },
    "GMT-1": {
      "exemplarCity": "GMT-1"
    },
    "GMT-10": {
      "exemplarCity": "GMT-10"
    },
    "GMT-11": {
      "exemplarCity": "GMT-11"
    },
    "GMT-12": {
      "exemplarCity": "GMT-12"
    },
    "GMT-13": {
      "exemplarCity": "GMT-13"
    },
    "GMT-14": {
      "exemplarCity": "GMT-14"
    },
    "GMT-2": {
      "exemplarCity": "GMT-2"
    },
    "GMT-3": {
      "exemplarCity": "GMT-3"
    },
    "GMT-4": {
      "exemplarCity": "GMT-4"
    },
    "GMT-5": {
      "exemplarCity": "GMT-5"
    },
    "GMT-6": {
      "exemplarCity": "GMT-6"
    },
    "GMT-7": {
      "exemplarCity": "GMT-7"
    },
    "GMT-8": {
      "exemplarCity": "GMT-8"
    },
    "GMT-9": {
      "exemplarCity": "GMT-9"
    },
    "Unknown": {
      "exemplarCity": "مدينة غير معروفة"
    }
  }
},

```



```

"metazone": {
  "Afghanistan": {
    "long": {
      "standard": "توقيت أفغانستان"
    }
  },
  "Africa_Central": {
    "long": {
      "standard": "توقيت وسط أفريقيا"
    }
  },
  "Africa_Eastern": {
    "long": {
      "standard": "توقيت شرق أفريقيا"
    }
  },
  "Africa_Southern": {
    "long": {
      "standard": "توقيت جنوب أفريقيا"
    }
  },
  "Africa_Western": {
    "long": {
      "generic": "توقيت غرب أفريقيا",
      "standard": "توقيت غرب أفريقيا الرسمي",
      "daylight": "توقيت غرب أفريقيا الصيفي"
    }
  },
  "Alaska": {
    "long": {
      "generic": "توقيت ألاسكا",
      "standard": "التوقيت الرسمي لألاسكا",
      "daylight": "توقيت ألاسكا الصيفي"
    }
  },
  "Amazon": {
    "long": {
      "generic": "توقيت الأمازون",
      "standard": "توقيت الأمازون الرسمي",
      "daylight": "توقيت الأمازون الصيفي"
    }
  },
  "America_Central": {
    "long": {
      "generic": "التوقيت المركزي لأمريكا الشمالية",
      "standard": "التوقيت الرسمي المركزي لأمريكا الشمالية",
      "daylight": "التوقيت الصيفي المركزي لأمريكا الشمالية"
    }
  },
  "America_Eastern": {
    "long": {
      "generic": "التوقيت الشرقي لأمريكا الشمالية",
      "standard": "التوقيت الرسمي الشرقي لأمريكا الشمالية",
      "daylight": "التوقيت الصيفي الشرقي لأمريكا الشمالية"
    }
  },
  "America_Mountain": {

```

```

    "long": {
      "generic": "التوقيت الجبلي لأمريكا الشمالية",
      "standard": "التوقيت الجبلي الرسمي لأمريكا الشمالية",
      "daylight": "التوقيت الجبلي الصيفي لأمريكا الشمالية"
    }
  },
  "America_Pacific": {
    "long": {
      "generic": "توقيت المحيط الهادي",
      "standard": "توقيت المحيط الهادي الرسمي",
      "daylight": "توقيت المحيط الهادي الصيفي"
    }
  },
  "Anadyr": {
    "long": {
      "generic": "توقيت أنادير",
      "standard": "توقيت أنادير الرسمي",
      "daylight": "التوقيت الصيفي لأنادير"
    }
  },
  "Apia": {
    "long": {
      "generic": "توقيت آبيا",
      "standard": "التوقيت الرسمي لآبيا",
      "daylight": "التوقيت الصيفي لآبيا"
    }
  },
  "Arabian": {
    "long": {
      "generic": "التوقيت العربي",
      "standard": "التوقيت العربي الرسمي",
      "daylight": "التوقيت العربي الصيفي"
    }
  },
  "Argentina": {
    "long": {
      "generic": "توقيت الأرجنتين",
      "standard": "توقيت الأرجنتين الرسمي",
      "daylight": "توقيت الأرجنتين الصيفي"
    }
  },
  "Argentina_Western": {
    "long": {
      "generic": "توقيت غرب الأرجنتين",
      "standard": "توقيت غرب الأرجنتين الرسمي",
      "daylight": "توقيت غرب الأرجنتين الصيفي"
    }
  },
  "Armenia": {
    "long": {
      "generic": "توقيت أرمينيا",
      "standard": "توقيت أرمينيا الرسمي",
      "daylight": "توقيت أرمينيا الصيفي"
    }
  },
  "Atlantic": {
    "long": {

```

```

        "generic": "توقيت الأطلسي",
        "standard": "التوقيت الرسمي الأطلسي",
        "daylight": "التوقيت الصيفي الأطلسي"
    },
},
"Australia_Central": {
    "long": {
        "generic": "توقيت وسط أستراليا",
        "standard": "توقيت وسط أستراليا الرسمي",
        "daylight": "توقيت وسط أستراليا الصيفي"
    }
},
"Australia_CentralWestern": {
    "long": {
        "generic": "توقيت غرب وسط أستراليا",
        "standard": "توقيت غرب وسط أستراليا الرسمي",
        "daylight": "توقيت غرب وسط أستراليا الصيفي"
    }
},
"Australia_Eastern": {
    "long": {
        "generic": "توقيت شرق أستراليا",
        "standard": "توقيت شرق أستراليا الرسمي",
        "daylight": "توقيت شرق أستراليا الصيفي"
    }
},
"Australia_Western": {
    "long": {
        "generic": "توقيت غرب أستراليا",
        "standard": "توقيت غرب أستراليا الرسمي",
        "daylight": "توقيت غرب أستراليا الصيفي"
    }
},
},
"Azerbaijan": {
    "long": {
        "generic": "توقيت أذربيجان",
        "standard": "توقيت أذربيجان الرسمي",
        "daylight": "توقيت أذربيجان الصيفي"
    }
},
},
"Azores": {
    "long": {
        "generic": "توقيت أزورس",
        "standard": "توقيت أزورس الرسمي",
        "daylight": "توقيت أزورس الصيفي"
    }
},
},
"Bangladesh": {
    "long": {
        "generic": "توقيت بنجلاديش",
        "standard": "توقيت بنجلاديش الرسمي",
        "daylight": "توقيت بنجلاديش الصيفي"
    }
},
},
"Bhutan": {
    "long": {
        "standard": "توقيت بوتان"
    }
}

```

```

    }
  },
  "Bolivia": {
    "long": {
      "standard": "توقيت بوليفيا"
    }
  },
  "Brasilia": {
    "long": {
      "generic": "توقيت برازيليا",
      "standard": "توقيت برازيليا الرسمي",
      "daylight": "توقيت برازيليا الصيفي"
    }
  },
  "Brunei": {
    "long": {
      "standard": "توقيت بروناي"
    }
  },
  "Cape_Verde": {
    "long": {
      "generic": "توقيت الرأس الأخضر",
      "standard": "توقيت الرأس الأخضر الرسمي",
      "daylight": "توقيت الرأس الأخضر الصيفي"
    }
  },
  "Chamorro": {
    "long": {
      "standard": "توقيت تشامورو"
    }
  },
  "Chatham": {
    "long": {
      "generic": "توقيت تشاتام",
      "standard": "توقيت تشاتام الرسمي",
      "daylight": "توقيت تشاتام الصيفي"
    }
  },
  "Chile": {
    "long": {
      "generic": "توقيت شيلي",
      "standard": "توقيت شيلي الرسمي",
      "daylight": "توقيت شيلي الصيفي"
    }
  },
  "China": {
    "long": {
      "generic": "توقيت الصين",
      "standard": "توقيت الصين الرسمي",
      "daylight": "توقيت الصين الصيفي"
    }
  },
  "Choibalsan": {
    "long": {
      "generic": "توقيت شويبالسان",
      "standard": "توقيت شويبالسان الرسمي",
      "daylight": "التوقيت الصيفي لشويبالسان"
    }
  }
}

```

```

    }
  },
  "Christmas": {
    "long": {
      "standard": "توقيت جزر الكريسماس"
    }
  },
  "Cocos": {
    "long": {
      "standard": "توقيت جزر كوكوس"
    }
  },
  "Colombia": {
    "long": {
      "generic": "توقيت كولومبيا",
      "standard": "توقيت كولومبيا الرسمي",
      "daylight": "توقيت كولومبيا الصيفي"
    }
  },
  "Cook": {
    "long": {
      "generic": "توقيت جزر كوك",
      "standard": "توقيت جزر كوك الرسمي",
      "daylight": "توقيت جزر كوك الصيفي"
    }
  },
  "Cuba": {
    "long": {
      "generic": "توقيت كوبا",
      "standard": "توقيت كوبا الرسمي",
      "daylight": "توقيت كوبا الصيفي"
    }
  },
  "Davis": {
    "long": {
      "standard": "توقيت دافيز"
    }
  },
  "DumontDUrville": {
    "long": {
      "standard": "توقيت دي مونت دو روفيل"
    }
  },
  "East_Timor": {
    "long": {
      "standard": "توقيت تيمور الشرقية"
    }
  },
  "Easter": {
    "long": {
      "generic": "توقيت جزيرة استر",
      "standard": "توقيت جزيرة استر الرسمي",
      "daylight": "توقيت جزيرة استر الصيفي"
    }
  },
  "Ecuador": {
    "long": {

```

```

        "standard": "توقيت الإكوادور"
    },
    },
    "Europe_Central": {
        "long": {
            "generic": "توقيت وسط أوروبا",
            "standard": "توقيت وسط أوروبا الرسمي",
            "daylight": "توقيت وسط أوروبا الصيفي"
        }
    },
    "Europe_Eastern": {
        "long": {
            "generic": "توقيت شرق أوروبا",
            "standard": "توقيت شرق أوروبا الرسمي",
            "daylight": "توقيت شرق أوروبا الصيفي"
        }
    },
    "Europe_Further_Eastern": {
        "long": {
            "standard": "التوقيت الأوروبي (أكثر شرقًا)"
        }
    },
    "Europe_Western": {
        "long": {
            "generic": "توقيت غرب أوروبا",
            "standard": "توقيت غرب أوروبا الرسمي",
            "daylight": "توقيت غرب أوروبا الصيفي"
        }
    },
    "Falkland": {
        "long": {
            "generic": "توقيت جزر فوكلاند",
            "standard": "توقيت جزر فوكلاند الرسمي",
            "daylight": "توقيت جزر فوكلاند الصيفي"
        }
    },
    "Fiji": {
        "long": {
            "generic": "توقيت فيجي",
            "standard": "توقيت فيجي الرسمي",
            "daylight": "توقيت فيجي الصيفي"
        }
    },
    "French_Guiana": {
        "long": {
            "standard": "توقيت غايانا الفرنسية"
        }
    },
    "French_Southern": {
        "long": {
            "standard": "توقيت المقاطعات الفرنسية الجنوبية والانتارتيكية"
        }
    },
    "Galapagos": {
        "long": {
            "standard": "توقيت غلاباغوس"
        }
    }
}

```

```

    },
    "Gambier": {
      "long": {
        "standard": "توقيت جامبير"
      }
    },
    "Georgia": {
      "long": {
        "generic": "توقيت جورجيا",
        "standard": "توقيت جورجيا الرسمي",
        "daylight": "توقيت جورجيا الصيفي"
      }
    },
    "Gilbert_Islands": {
      "long": {
        "standard": "توقيت جزر جيلبرت"
      }
    },
    "GMT": {
      "long": {
        "standard": "توقيت غرينتش"
      }
    },
    "Greenland_Eastern": {
      "long": {
        "generic": "توقيت شرق غرينلاند",
        "standard": "توقيت شرق غرينلاند الرسمي",
        "daylight": "توقيت شرق غرينلاند الصيفي"
      }
    },
    "Greenland_Western": {
      "long": {
        "generic": "توقيت غرب غرينلاند",
        "standard": "توقيت غرب غرينلاند الرسمي",
        "daylight": "توقيت غرب غرينلاند الصيفي"
      }
    },
    "Guam": {
      "long": {
        "standard": "توقيت غوام"
      }
    },
    "Gulf": {
      "long": {
        "standard": "توقيت الخليج"
      }
    },
    "Guyana": {
      "long": {
        "standard": "توقيت غيانا"
      }
    },
    "Hawaii_Aleutian": {
      "long": {
        "generic": "توقيت هاواي ألوتيان",
        "standard": "توقيت هاواي ألوتيان الرسمي",
        "daylight": "توقيت هاواي ألوتيان الصيفي"
      }
    }
  }

```

```

    }
  },
  "Hong_Kong": {
    "long": {
      "generic": "توقيت هونغ كونغ",
      "standard": "توقيت هونغ كونغ الرسمي",
      "daylight": "توقيت هونغ كونغ الصيفي"
    }
  },
  "Hovd": {
    "long": {
      "generic": "توقيت هوفد",
      "standard": "توقيت هوفد الرسمي",
      "daylight": "توقيت هوفد الصيفي"
    }
  },
  "India": {
    "long": {
      "standard": "توقيت الهند"
    }
  },
  "Indian_Ocean": {
    "long": {
      "standard": "توقيت المحيط الهندي"
    }
  },
  "Indochina": {
    "long": {
      "standard": "توقيت الهند الصينية"
    }
  },
  "Indonesia_Central": {
    "long": {
      "standard": "توقيت وسط إندونيسيا"
    }
  },
  "Indonesia_Eastern": {
    "long": {
      "standard": "توقيت شرق إندونيسيا"
    }
  },
  "Indonesia_Western": {
    "long": {
      "standard": "توقيت غرب إندونيسيا"
    }
  },
  "Iran": {
    "long": {
      "generic": "توقيت إيران",
      "standard": "توقيت إيران الرسمي",
      "daylight": "توقيت إيران الصيفي"
    }
  },
  "Irkutsk": {
    "long": {
      "generic": "توقيت إركوتسك",
      "standard": "توقيت إركوتسك الرسمي",

```



```

        "daylight": "توقيت إركوتسك الصيفي"
    },
},
"Israel": {
    "long": {
        "generic": "توقيت إسرائيل",
        "standard": "توقيت إسرائيل الرسمي",
        "daylight": "توقيت إسرائيل الصيفي"
    }
},
"Japan": {
    "long": {
        "generic": "توقيت اليابان",
        "standard": "توقيت اليابان الرسمي",
        "daylight": "توقيت اليابان الصيفي"
    }
},
"Kamchatka": {
    "long": {
        "generic": "توقيت كامشاتكا",
        "standard": "توقيت بيتروبافلوفسك-كامتشاتسكي",
        "daylight": "توقيت بيتروبافلوفسك-كامتشاتسكي الصيفي"
    }
},
"Kazakhstan_Eastern": {
    "long": {
        "standard": "توقيت شرق كازاخستان"
    }
},
"Kazakhstan_Western": {
    "long": {
        "standard": "توقيت غرب كازاخستان"
    }
},
"Korea": {
    "long": {
        "generic": "توقيت كوريا",
        "standard": "توقيت كوريا الرسمي",
        "daylight": "توقيت كوريا الصيفي"
    }
},
"Kosrae": {
    "long": {
        "standard": "توقيت كوسرا"
    }
},
"Krasnoyarsk": {
    "long": {
        "generic": "توقيت كراسنويارسك",
        "standard": "توقيت كراسنويارسك الرسمي",
        "daylight": "التوقيت الصيفي لكراسنويارسك"
    }
},
"Kyrgystan": {
    "long": {
        "standard": "توقيت قرغيزستان"
    }
}

```

```

    },
    "Line_Islands": {
      "long": {
        "standard": "توقيت جزر لاین"
      }
    },
    "Lord_Howe": {
      "long": {
        "generic": "توقيت لورد هاو",
        "standard": "توقيت لورد هاو الرسمي",
        "daylight": "التوقيت الصيفي للورد هاو"
      }
    },
    "Macquarie": {
      "long": {
        "standard": "توقيت ماكوارى"
      }
    },
    "Magadan": {
      "long": {
        "generic": "توقيت ماغادان",
        "standard": "توقيت ماغادان الرسمي",
        "daylight": "توقيت ماغادان الصيفي"
      }
    },
    "Malaysia": {
      "long": {
        "standard": "توقيت ماليزيا"
      }
    },
    "Maldives": {
      "long": {
        "standard": "توقيت المالديف"
      }
    },
    "Marquesas": {
      "long": {
        "standard": "توقيت ماركيساس"
      }
    },
    "Marshall_Islands": {
      "long": {
        "standard": "توقيت جزر مارشال"
      }
    },
    "Mauritius": {
      "long": {
        "generic": "توقيت موريشيوس",
        "standard": "توقيت موريشيوس الرسمي",
        "daylight": "توقيت موريشيوس الصيفي"
      }
    },
    "Mawson": {
      "long": {
        "standard": "توقيت ماوسون"
      }
    }
  },

```

```

"Mexico_Northwest": {
  "long": {
    "generic": "توقيت شمال غرب المكسيك",
    "standard": "التوقيت الرسمي لشمال غرب المكسيك",
    "daylight": "التوقيت الصيفي لشمال غرب المكسيك"
  }
},
"Mexico_Pacific": {
  "long": {
    "generic": "توقيت المحيط الهادي للمكسيك",
    "standard": "توقيت المحيط الهادي الرسمي للمكسيك",
    "daylight": "توقيت المحيط الهادي الصيفي للمكسيك"
  }
},
"Mongolia": {
  "long": {
    "generic": "توقيت أولان باتور",
    "standard": "توقيت أولان باتور الرسمي",
    "daylight": "توقيت أولان باتور الصيفي"
  }
},
"Moscow": {
  "long": {
    "generic": "توقيت موسكو",
    "standard": "توقيت موسكو الرسمي",
    "daylight": "توقيت موسكو الصيفي"
  }
},
"Myanmar": {
  "long": {
    "standard": "توقيت ميانمار"
  }
},
"Nauru": {
  "long": {
    "standard": "توقيت ناورو"
  }
},
"Nepal": {
  "long": {
    "standard": "توقيت نيبال"
  }
},
"New_Caledonia": {
  "long": {
    "generic": "توقيت كاليدونيا الجديدة",
    "standard": "توقيت كاليدونيا الجديدة الرسمي",
    "daylight": "توقيت كاليدونيا الجديدة الصيفي"
  }
},
"New_Zealand": {
  "long": {
    "generic": "توقيت نيوزيلندا",
    "standard": "توقيت نيوزيلندا الرسمي",
    "daylight": "توقيت نيوزيلندا الصيفي"
  }
},

```

```

    "Newfoundland": {
      "long": {
        "generic": "توقيت نيوفاوندلاند",
        "standard": "توقيت نيوفاوندلاند الرسمي",
        "daylight": "توقيت نيوفاوندلاند الصيفي"
      }
    },
    "Niue": {
      "long": {
        "standard": "توقيت نيوي"
      }
    },
    "Norfolk": {
      "long": {
        "standard": "توقيت جزيرة نورفولك"
      }
    },
    "Noronha": {
      "long": {
        "generic": "توقيت فيرناندو دي نورونها",
        "standard": "توقيت فرناندو دي نورونها الرسمي",
        "daylight": "توقيت فرناندو دي نورونها الصيفي"
      }
    },
    "North_Mariana": {
      "long": {
        "standard": "توقيت جزر ماريانا الشمالية"
      }
    },
    "Novosibirsk": {
      "long": {
        "generic": "توقيت نوفوسيبيرسك",
        "standard": "توقيت نوفوسيبيرسك الرسمي",
        "daylight": "توقيت نوفوسيبيرسك الصيفي"
      }
    },
    "Omsk": {
      "long": {
        "generic": "توقيت أومسك",
        "standard": "توقيت أومسك الرسمي",
        "daylight": "توقيت أومسك الصيفي"
      }
    },
    "Pakistan": {
      "long": {
        "generic": "توقيت باكستان",
        "standard": "توقيت باكستان الرسمي",
        "daylight": "توقيت باكستان الصيفي"
      }
    },
    "Palau": {
      "long": {
        "standard": "توقيت بالاو"
      }
    },
    "Papua_New_Guinea": {
      "long": {

```

```

        "standard": "توقيت بابوا غينيا الجديدة"
    },
    },
    "Paraguay": {
        "long": {
            "generic": "توقيت باراغواي",
            "standard": "توقيت باراغواي الرسمي",
            "daylight": "توقيت باراغواي الصيفي"
        }
    },
    "Peru": {
        "long": {
            "generic": "توقيت بيرو",
            "standard": "توقيت بيرو الرسمي",
            "daylight": "توقيت بيرو الصيفي"
        }
    },
    "Philippines": {
        "long": {
            "generic": "توقيت الفلبين",
            "standard": "توقيت الفلبين الرسمي",
            "daylight": "توقيت الفلبين الصيفي"
        }
    },
    "Phoenix_Islands": {
        "long": {
            "standard": "توقيت جزر فينكس"
        }
    },
    "Pierre_Miquelon": {
        "long": {
            "generic": "توقيت سانت بيير وميكلون",
            "standard": "توقيت سانت بيير وميكلون الرسمي",
            "daylight": "توقيت سانت بيير وميكلون الصيفي"
        }
    },
    "Pitcairn": {
        "long": {
            "standard": "توقيت بيتكيرن"
        }
    },
    "Ponape": {
        "long": {
            "standard": "توقيت بونابي"
        }
    },
    "Pyongyang": {
        "long": {
            "standard": "توقيت بيونغ يانغ"
        }
    },
    "Reunion": {
        "long": {
            "standard": "توقيت ريونيون"
        }
    },
    "Rothera": {

```

```

        "long": {
            "standard": "توقيت روثيرا"
        }
    },
    "Sakhalin": {
        "long": {
            "generic": "توقيت ساخالين",
            "standard": "توقيت ساخالين الرسمي",
            "daylight": "توقيت ساخالين الصيفي"
        }
    },
    "Samara": {
        "long": {
            "generic": "توقيت سامارا",
            "standard": "توقيت سامارا",
            "daylight": "توقيت سامارا الصيفي"
        }
    },
    "Samoa": {
        "long": {
            "generic": "توقيت ساموا",
            "standard": "توقيت ساموا الرسمي",
            "daylight": "توقيت ساموا الصيفي"
        }
    },
    "Seychelles": {
        "long": {
            "standard": "توقيت سيشل"
        }
    },
    "Singapore": {
        "long": {
            "standard": "توقيت سنغافورة"
        }
    },
    "Solomon": {
        "long": {
            "standard": "توقيت جزر سليمان"
        }
    },
    "South_Georgia": {
        "long": {
            "standard": "توقيت جنوب جورجيا"
        }
    },
    "Suriname": {
        "long": {
            "standard": "توقيت سورينام"
        }
    },
    "Syowa": {
        "long": {
            "standard": "توقيت سايووا"
        }
    },
    "Tahiti": {
        "long": {

```

```

        "standard": "توقيت تاهيتي"
    },
    },
    "Taipei": {
        "long": {
            "generic": "توقيت تايبيه",
            "standard": "توقيت تايبيه الرسمي",
            "daylight": "توقيت تايبيه الصيفي"
        }
    },
    "Tajikistan": {
        "long": {
            "standard": "توقيت طاجكستان"
        }
    },
    "Tokelau": {
        "long": {
            "standard": "توقيت توكيلاو"
        }
    },
    },
    "Tonga": {
        "long": {
            "generic": "توقيت تونغا",
            "standard": "توقيت تونغا الرسمي",
            "daylight": "توقيت تونغا الصيفي"
        }
    },
    },
    "Truk": {
        "long": {
            "standard": "توقيت شوك"
        }
    },
    },
    "Turkmenistan": {
        "long": {
            "generic": "توقيت تركمانستان",
            "standard": "توقيت تركمانستان الرسمي",
            "daylight": "توقيت تركمانستان الصيفي"
        }
    },
    },
    "Tuvalu": {
        "long": {
            "standard": "توقيت توفالو"
        }
    },
    },
    "Uruguay": {
        "long": {
            "generic": "توقيت أورغواي",
            "standard": "توقيت أورغواي الرسمي",
            "daylight": "توقيت أورغواي الصيفي"
        }
    },
    },
    "Uzbekistan": {
        "long": {
            "generic": "توقيت أوزبكستان",
            "standard": "توقيت أوزبكستان الرسمي",
            "daylight": "توقيت أوزبكستان الصيفي"
        }
    }
}

```

```

    },
    "Vanuatu": {
      "long": {
        "generic": "توقيت فانواتو",
        "standard": "توقيت فانواتو الرسمي",
        "daylight": "توقيت فانواتو الصيفي"
      }
    },
    "Venezuela": {
      "long": {
        "standard": "توقيت فنزويلا"
      }
    },
    "Vladivostok": {
      "long": {
        "generic": "توقيت فلاديفوستوك",
        "standard": "توقيت فلاديفوستوك الرسمي",
        "daylight": "توقيت فلاديفوستوك الصيفي"
      }
    },
    "Volgograd": {
      "long": {
        "generic": "توقيت فولغوغراد",
        "standard": "توقيت فولغوغراد الرسمي",
        "daylight": "توقيت فولغوغراد الصيفي"
      }
    },
    "Vostok": {
      "long": {
        "standard": "توقيت فوستوك"
      }
    },
    "Wake": {
      "long": {
        "standard": "توقيت جزيرة ويك"
      }
    },
    "Wallis": {
      "long": {
        "standard": "توقيت واليس و فوتونا"
      }
    },
    "Yakutsk": {
      "long": {
        "generic": "توقيت ياكوتسك",
        "standard": "توقيت ياكوتسك الرسمي",
        "daylight": "توقيت ياكوتسك الصيفي"
      }
    },
    "Yekaterinburg": {
      "long": {
        "generic": "توقيت يكاترينبورغ",
        "standard": "توقيت يكاترينبورغ الرسمي",
        "daylight": "توقيت يكاترينبورغ الصيفي"
      }
    }
  }
}

```



```

    }
  }
}
}
}

```

TIMEZONENAMES.JSX

```

{
  "main";
  {
    "ar";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 12879 $",
          "_cldrVersion";
          "30.0.3";
        }
        "language";
        "ar";
      }
    }
    "dates";
    {
      "timeZoneNames";
      {
        "hourFormat";
        "+HH:mm;-HH:mm",
        "gmtFormat";
        "0{جرينتش",
        "gmtZeroFormat";
        "جرينتش",
        "regionFormat";
        "0{توقيت",
        "regionFormat-type-daylight";
        "الصيفي} 0{توقيت",
        "regionFormat-type-standard";
        "الرسمي} 0{توقيت",
        "fallbackFormat";
        "{1} ({0}) ",
        "zone";
        {
          "America";
          {
            "Adak";
            {
              "exemplarCity";
              "أداك";
            }
            "Anchorage";
            {
              "exemplarCity";
              "أنشوراج";
            }
          }
        }
      }
    }
  }
}

```

```
}
"Anguilla";
{
  "exemplarCity";
  "أنغيلا";
}
"Antigua";
{
  "exemplarCity";
  "أنتيغوا";
}
"Araguaina";
{
  "exemplarCity";
  "أروجوانيا";
}
"Argentina";
{
  "Rio_Gallegos";
  {
    "exemplarCity";
    "ريو جالييوس";
  }
  "San_Juan";
  {
    "exemplarCity";
    "سان خوان";
  }
  "Ushuaia";
  {
    "exemplarCity";
    "أشوا";
  }
  "La_Rioja";
  {
    "exemplarCity";
    "لا ريوجا";
  }
  "San_Luis";
  {
    "exemplarCity";
    "سان لويس";
  }
  "Salta";
  {
    "exemplarCity";
    "سالطا";
  }
  "Tucuman";
  {
    "exemplarCity";
    "تاكمان";
  }
}
"Aruba";
{
  "exemplarCity";
```

```

        "أروبا";
    }
    "Asuncion";
    {
        "exemplarCity";
        "أسونسيون";
    }
    "Bahia";
    {
        "exemplarCity";
        "باهيا";
    }
    "Bahia_Banderas";
    {
        "exemplarCity";
        "باهيا بانديراس";
    }
    "Barbados";
    {
        "exemplarCity";
        "بربادوس";
    }
    "Belem";
    {
        "exemplarCity";
        "بلم";
    }
    "Belize";
    {
        "exemplarCity";
        "بليز";
    }
    "Blanc-Sablon";
    {
        "exemplarCity";
        "بلانك-سابلون";
    }
    "Boa_Vista";
    {
        "exemplarCity";
        "باو فيستا";
    }
    "Bogota";
    {
        "exemplarCity";
        "بوغوتا";
    }
    "Boise";
    {
        "exemplarCity";
        "بويس";
    }
    "Buenos_Aires";
    {
        "exemplarCity";
        "بوينوس أيرس";
    }
}

```

```
"Cambridge_Bay";
{
  "exemplarCity";
  "کامبردیج باي";
}
"Campo_Grande";
{
  "exemplarCity";
  "کومبو جراند";
}
"Cancun";
{
  "exemplarCity";
  "کانکون";
}
"Caracas";
{
  "exemplarCity";
  "کاراکاس";
}
"Catamarca";
{
  "exemplarCity";
  "کاتامارکا";
}
"Cayenne";
{
  "exemplarCity";
  "کایین";
}
"Cayman";
{
  "exemplarCity";
  "کیمان";
}
"Chicago";
{
  "exemplarCity";
  "شیکاگو";
}
"Chihuahua";
{
  "exemplarCity";
  "تشیواوا";
}
"Coral_Harbour";
{
  "exemplarCity";
  "کورانال هاربر";
}
"Cordoba";
{
  "exemplarCity";
  "کوردوبا";
}
"Costa_Rica";
{
```

```

        "exemplarCity";
        "كوستاريكا";
    }
    "Creston";
    {
        "exemplarCity";
        "كريستون";
    }
    "Cuiaba";
    {
        "exemplarCity";
        "كيابا";
    }
    "Curacao";
    {
        "exemplarCity";
        "كوراكاو";
    }
    "Danmarkshavn";
    {
        "exemplarCity";
        "دانمرك شافن";
    }
    "Dawson";
    {
        "exemplarCity";
        "داوسان";
    }
    "Dawson_Creek";
    {
        "exemplarCity";
        "داوسن كريك";
    }
    "Denver";
    {
        "exemplarCity";
        "دنفر";
    }
    "Detroit";
    {
        "exemplarCity";
        "ديترويت";
    }
    "Dominica";
    {
        "exemplarCity";
        "دومينيكا";
    }
    "Edmonton";
    {
        "exemplarCity";
        "ايدمونتون";
    }
    "Eirunepe";
    {
        "exemplarCity";
        "ايرونبي";
    }

```

```
}
"El_Salvador";
{
  "exemplarCity";
  "السلفادور";
}
"Fort_Nelson";
{
  "exemplarCity";
  "فورت نيلسون";
}
"Fortaleza";
{
  "exemplarCity";
  "فورتاليزا";
}
"Glace_Bay";
{
  "exemplarCity";
  "جلاس باي";
}
"Godthab";
{
  "exemplarCity";
  "غودثاب";
}
"Goose_Bay";
{
  "exemplarCity";
  "جوس باي";
}
"Grand_Turk";
{
  "exemplarCity";
  "گرانند ترك";
}
"Grenada";
{
  "exemplarCity";
  "غرينادا";
}
"Guadeloupe";
{
  "exemplarCity";
  "غوادلوب";
}
"Guatemala";
{
  "exemplarCity";
  "غواتيمالا";
}
"Guayaquil";
{
  "exemplarCity";
  "غواياكويل";
}
"Guyana";
```

```

    {
      "exemplarCity";
      "غيانا";
    }
    "Halifax";
    {
      "exemplarCity";
      "هاليفاكس";
    }
    "Havana";
    {
      "exemplarCity";
      "هافانا";
    }
    "Hermosillo";
    {
      "exemplarCity";
      "هيرموسيلو";
    }
    "Indiana";
    {
      "Vincennes";
      {
        "exemplarCity";
        "فينسينس";
      }
      "Petersburg";
      {
        "exemplarCity";
        "بيتربيرغ";
      }
      "Tell_City";
      {
        "exemplarCity";
        "مدينة تل، إنديانا";
      }
      "Knox";
      {
        "exemplarCity";
        "كونكس";
      }
      "Winamac";
      {
        "exemplarCity";
        "ويناماك";
      }
      "Marengo";
      {
        "exemplarCity";
        "مارنجو";
      }
      "Vevay";
      {
        "exemplarCity";
        "فيفاي";
      }
    }
  }

```

```

"Indianapolis";
{
  "exemplarCity";
  "إنديانا بوليس";
}
"Inuvik";
{
  "exemplarCity";
  "اينوفيك";
}
"Iqaluit";
{
  "exemplarCity";
  "اكويلت";
}
"Jamaica";
{
  "exemplarCity";
  "جاما يكا";
}
"Jujuy";
{
  "exemplarCity";
  "جوجو";
}
"Juneau";
{
  "exemplarCity";
  "جونى";
}
"Kentucky";
{
  "Monticello";
  {
    "exemplarCity";
    "مونتي سيلو";
  }
}
"Kralendijk";
{
  "exemplarCity";
  "كرالنديك";
}
"La_Paz";
{
  "exemplarCity";
  "لا باز";
}
"Lima";
{
  "exemplarCity";
  "ليما";
}
"Los_Angeles";
{
  "exemplarCity";
  "لوس انجلوس";
}

```



```
}
"Louisville";
{
  "exemplarCity";
  "لويس فيل";
}
"Lower_Princes";
{
  "exemplarCity";
  "حي الأمير السفلي";
}
"Maceio";
{
  "exemplarCity";
  "ماشيو";
}
"Managua";
{
  "exemplarCity";
  "ماناغوا";
}
"Manaus";
{
  "exemplarCity";
  "ماناوس";
}
"Marigot";
{
  "exemplarCity";
  "ماريغوت";
}
"Martinique";
{
  "exemplarCity";
  "المارتينيك";
}
"Matamoros";
{
  "exemplarCity";
  "ماتاموروس";
}
"Mazatlan";
{
  "exemplarCity";
  "مازاتلان";
}
"Mendoza";
{
  "exemplarCity";
  "ميندوزا";
}
"Menominee";
{
  "exemplarCity";
  "مينوميني";
}
"Merida";
```

```

{
    "exemplarCity";
    "ميريدا";
}
"Metlakatla";
{
    "exemplarCity";
    "ميتلاكاتلا";
}
"Mexico_City";
{
    "exemplarCity";
    "مدينة المكسيك";
}
"Miquelon";
{
    "exemplarCity";
    "ميكلون";
}
"Moncton";
{
    "exemplarCity";
    "وينكتون";
}
"Monterrey";
{
    "exemplarCity";
    "مونتيري";
}
"Montevideo";
{
    "exemplarCity";
    "مونتيفيديو";
}
"Montserrat";
{
    "exemplarCity";
    "مونتسيرات";
}
"Nassau";
{
    "exemplarCity";
    "ناسو";
}
"New_York";
{
    "exemplarCity";
    "نيويورك";
}
"Nipigon";
{
    "exemplarCity";
    "نبيجون";
}
"Nome";
{
    "exemplarCity";

```

```

        "نوم";
    }
    "Noronha";
    {
        "exemplarCity";
        "نوروناه";
    }
    "North_Dakota";
    {
        "Beulah";
        {
            "exemplarCity";
            "بيولا، داكوتا الشمالية";
        }
        "New_Salem";
        {
            "exemplarCity";
            "نيو ساليم";
        }
        "Center";
        {
            "exemplarCity";
            "سنتر";
        }
    }
    "Ojinaga";
    {
        "exemplarCity";
        "أوجيناجا";
    }
    "Panama";
    {
        "exemplarCity";
        "بنما";
    }
    "Pangnirtung";
    {
        "exemplarCity";
        "بانجينتيج";
    }
    "Paramaribo";
    {
        "exemplarCity";
        "باراماريبو";
    }
    "Phoenix";
    {
        "exemplarCity";
        "فينكس";
    }
    "Port-au-Prince";
    {
        "exemplarCity";
        "بورت أو برنس";
    }
    "Port_of_Spain";
    {

```

```

        "exemplarCity";
        "بورت أوف سبين";
    }
    "Porto_Velho";
    {
        "exemplarCity";
        "بورتو فيلو";
    }
    "Puerto_Rico";
    {
        "exemplarCity";
        "بورتوريكو";
    }
    "Rainy_River";
    {
        "exemplarCity";
        "راني ريفر";
    }
    "Rankin_Inlet";
    {
        "exemplarCity";
        "رانكن انلت";
    }
    "Recife";
    {
        "exemplarCity";
        "ريسيف";
    }
    "Regina";
    {
        "exemplarCity";
        "ريجينا";
    }
    "Resolute";
    {
        "exemplarCity";
        "ريزولوت";
    }
    "Rio_Branco";
    {
        "exemplarCity";
        "ريوبرانكو";
    }
    "Santa_Isabel";
    {
        "exemplarCity";
        "سانتا إيزابيل";
    }
    "Santarem";
    {
        "exemplarCity";
        "سانتاريم";
    }
    "Santiago";
    {
        "exemplarCity";
        "سانتياغو";
    }

```

```

    }
    "Santo_Domingo";
    {
        "exemplarCity";
        "سانتو دومينغو";
    }
    "Sao_Paulo";
    {
        "exemplarCity";
        "ساو باولو";
    }
    "Scoresbysund";
    {
        "exemplarCity";
        "سکورسبیسند";
    }
    "Sitka";
    {
        "exemplarCity";
        "سیتکا";
    }
    "St_Barthelemy";
    {
        "exemplarCity";
        "سانت بارتیلیمی";
    }
    "St_Johns";
    {
        "exemplarCity";
        "سانت جونز";
    }
    "St_Kitts";
    {
        "exemplarCity";
        "سانت کیتس";
    }
    "St_Lucia";
    {
        "exemplarCity";
        "سانت لوشیا";
    }
    "St_Thomas";
    {
        "exemplarCity";
        "سانت توماس";
    }
    "St_Vincent";
    {
        "exemplarCity";
        "سانت فنسنت";
    }
    "Swift_Current";
    {
        "exemplarCity";
        "سوفت کارنت";
    }
    "Tegucigalpa";

```

```

        {
            "exemplarCity";
            "تيغوسيغالبيا";
        }
        "Thule";
        {
            "exemplarCity";
            "ثيل";
        }
        "Thunder_Bay";
        {
            "exemplarCity";
            "ثندر باي";
        }
        "Tijuana";
        {
            "exemplarCity";
            "تيخوانا";
        }
        "Toronto";
        {
            "exemplarCity";
            "تورونتو";
        }
        "Tortola";
        {
            "exemplarCity";
            "تورتولا";
        }
        "Vancouver";
        {
            "exemplarCity";
            "فانكوفر";
        }
        "Whitehorse";
        {
            "exemplarCity";
            "وايت هورس";
        }
        "Winnipeg";
        {
            "exemplarCity";
            "وينيبيج";
        }
        "Yakutat";
        {
            "exemplarCity";
            "ياكوتات";
        }
        "Yellowknife";
        {
            "exemplarCity";
            "يلونيف";
        }
    }
    "Atlantic";
    {

```

```
"Azores";
{
  "exemplarCity";
  "أزورس";
}
"Bermuda";
{
  "exemplarCity";
  "برمودا";
}
"Canary";
{
  "exemplarCity";
  "كناري";
}
"Cape_Verde";
{
  "exemplarCity";
  "الرأس الأخضر";
}
"Faeroe";
{
  "exemplarCity";
  "فارو";
}
"Madeira";
{
  "exemplarCity";
  "ماديرا";
}
"Reykjavik";
{
  "exemplarCity";
  "ريكيافيك";
}
"South_Georgia";
{
  "exemplarCity";
  "جورجيا الجنوبية";
}
"St_Helena";
{
  "exemplarCity";
  "سانت هيلينا";
}
"Stanley";
{
  "exemplarCity";
  "استانلي";
}
}
"Europe";
{
  "Amsterdam";
  {
    "exemplarCity";
    "أمستردام";
  }
}
```

```
}
"Andorra";
{
  "exemplarCity";
  "أندورا";
}
"Astrakhan";
{
  "exemplarCity";
  "أستراخان";
}
"Athens";
{
  "exemplarCity";
  "أثينا";
}
"Belgrade";
{
  "exemplarCity";
  "بلغراد";
}
"Berlin";
{
  "exemplarCity";
  "برلين";
}
"Bratislava";
{
  "exemplarCity";
  "براتيسلافا";
}
"Brussels";
{
  "exemplarCity";
  "بروكسل";
}
"Bucharest";
{
  "exemplarCity";
  "بوخارست";
}
"Budapest";
{
  "exemplarCity";
  "بودابست";
}
"Busingen";
{
  "exemplarCity";
  "بوسنغن";
}
"Chisinau";
{
  "exemplarCity";
  "تشيسيناو";
}
"Copenhagen";
```



```

{
    "exemplarCity";
    "كوبنهاغن";
}
"Dublin";
{
    "long";
    {
        "daylight";
        "توقيت أيرلندا الرسمي";
    }
    "exemplarCity";
    "دبلن";
}
"Gibraltar";
{
    "exemplarCity";
    "جبل طارق";
}
"Guernsey";
{
    "exemplarCity";
    "غيرنسي";
}
"Helsinki";
{
    "exemplarCity";
    "هلسنكي";
}
"Isle_of_Man";
{
    "exemplarCity";
    "جزيرة مان";
}
"Istanbul";
{
    "exemplarCity";
    "إسطنبول";
}
"Jersey";
{
    "exemplarCity";
    "جيرسي";
}
"Kaliningrad";
{
    "exemplarCity";
    "كالينجراد";
}
"Kiev";
{
    "exemplarCity";
    "كييف";
}
"Kirov";
{
    "exemplarCity";

```

```

        "كيروف";
    }
    "Lisbon";
    {
        "exemplarCity";
        "لشبونة";
    }
    "Ljubljana";
    {
        "exemplarCity";
        "ليوبليانا";
    }
    "London";
    {
        "long";
        {
            "daylight";
            "توقيت بريطانيا الصيفي";
        }
        "exemplarCity";
        "لندن";
    }
    "Luxembourg";
    {
        "exemplarCity";
        "لوكسمبورغ";
    }
    "Madrid";
    {
        "exemplarCity";
        "مدريد";
    }
    "Malta";
    {
        "exemplarCity";
        "مالطة";
    }
    "Mariehamn";
    {
        "exemplarCity";
        "ماريهامن";
    }
    "Minsk";
    {
        "exemplarCity";
        "مينسك";
    }
    "Monaco";
    {
        "exemplarCity";
        "موناكو";
    }
    "Moscow";
    {
        "exemplarCity";
        "موسكو";
    }
}

```

```
"Oslo";
{
  "exemplarCity";
  "أوسلو";
}
"Paris";
{
  "exemplarCity";
  "باريس";
}
"Podgorica";
{
  "exemplarCity";
  "بودغوريكا";
}
"Prague";
{
  "exemplarCity";
  "براغ";
}
"Riga";
{
  "exemplarCity";
  "ريغا";
}
"Rome";
{
  "exemplarCity";
  "روما";
}
"Samara";
{
  "exemplarCity";
  "سمراء";
}
"San_Marino";
{
  "exemplarCity";
  "سان مارينو";
}
"Sarajevo";
{
  "exemplarCity";
  "سراييفو";
}
"Simferopol";
{
  "exemplarCity";
  "سيمفروبول";
}
"Skopje";
{
  "exemplarCity";
  "سكوبي";
}
"Sofia";
{
```

```

        "exemplarCity";
        "صوفيا";
    }
    "Stockholm";
    {
        "exemplarCity";
        "ستوكهولم";
    }
    "Tallinn";
    {
        "exemplarCity";
        "تالين";
    }
    "Tirane";
    {
        "exemplarCity";
        "تيرانا";
    }
    "Ulyanovsk";
    {
        "exemplarCity";
        "أوليانوفسك";
    }
    "Uzhgorod";
    {
        "exemplarCity";
        "أوزجروود";
    }
    "Vaduz";
    {
        "exemplarCity";
        "فادوز";
    }
    "Vatican";
    {
        "exemplarCity";
        "الفاتيكان";
    }
    "Vienna";
    {
        "exemplarCity";
        "فيينا";
    }
    "Vilnius";
    {
        "exemplarCity";
        "فيلنيوس";
    }
    "Volgograd";
    {
        "exemplarCity";
        "فولجograd";
    }
    "Warsaw";
    {
        "exemplarCity";
        "وارسو";
    }

```

```
}
  "Zagreb";
  {
    "exemplarCity";
    "زغرب";
  }
  "Zaporozhye";
  {
    "exemplarCity";
    "زابوروزي";
  }
  "Zurich";
  {
    "exemplarCity";
    "زيورخ";
  }
}
"Africa";
{
  "Abidjan";
  {
    "exemplarCity";
    "أبيدجان";
  }
  "Accra";
  {
    "exemplarCity";
    "أكرا";
  }
  "Addis_Ababa";
  {
    "exemplarCity";
    "أديس أبابا";
  }
  "Algiers";
  {
    "exemplarCity";
    "الجزائر";
  }
  "Asmera";
  {
    "exemplarCity";
    "أسمرّة";
  }
  "Bamako";
  {
    "exemplarCity";
    "باماكو";
  }
  "Bangui";
  {
    "exemplarCity";
    "بانغوي";
  }
  "Banjul";
  {
    "exemplarCity";
```

```

        "بانجول";
    }
    "Bissau";
    {
        "exemplarCity";
        "بيساو";
    }
    "Blantyre";
    {
        "exemplarCity";
        "بلانتيير";
    }
    "Brazzaville";
    {
        "exemplarCity";
        "برازافيل";
    }
    "Bujumbura";
    {
        "exemplarCity";
        "بوجومبورا";
    }
    "Cairo";
    {
        "exemplarCity";
        "القاهرة";
    }
    "Casablanca";
    {
        "exemplarCity";
        "الدار البيضاء";
    }
    "Ceuta";
    {
        "exemplarCity";
        "سيتا";
    }
    "Conakry";
    {
        "exemplarCity";
        "كوناكري";
    }
    "Dakar";
    {
        "exemplarCity";
        "داكار";
    }
    "Dar_es_Salaam";
    {
        "exemplarCity";
        "دار السلام";
    }
    "Djibouti";
    {
        "exemplarCity";
        "جيبوتي";
    }
}

```

```
"Douala";
{
  "exemplarCity";
  "دوالا";
}
"El_Aaiun";
{
  "exemplarCity";
  "العيون";
}
"Freetown";
{
  "exemplarCity";
  "فري تاون";
}
"Gaborone";
{
  "exemplarCity";
  "غابورون";
}
"Harare";
{
  "exemplarCity";
  "هراري";
}
"Johannesburg";
{
  "exemplarCity";
  "جوهانسبرغ";
}
"Juba";
{
  "exemplarCity";
  "جوبا";
}
"Kampala";
{
  "exemplarCity";
  "كامبالا";
}
"Khartoum";
{
  "exemplarCity";
  "الخرطوم";
}
"Kigali";
{
  "exemplarCity";
  "كيغالي";
}
"Kinshasa";
{
  "exemplarCity";
  "كينشاسا";
}
"Lagos";
{
```

```

        "exemplarCity";
        "لاغوس";
    }
    "Libreville";
    {
        "exemplarCity";
        "ليبرفيل";
    }
    "Lome";
    {
        "exemplarCity";
        "لومي";
    }
    "Luanda";
    {
        "exemplarCity";
        "لواندا";
    }
    "Lubumbashi";
    {
        "exemplarCity";
        "لومببasha";
    }
    "Lusaka";
    {
        "exemplarCity";
        "لوساكا";
    }
    "Malabo";
    {
        "exemplarCity";
        "مالابو";
    }
    "Maputo";
    {
        "exemplarCity";
        "مابوتو";
    }
    "Maseru";
    {
        "exemplarCity";
        "ماسيرو";
    }
    "Mbabane";
    {
        "exemplarCity";
        "مباباني";
    }
    "Mogadishu";
    {
        "exemplarCity";
        "مقديشو";
    }
    "Monrovia";
    {
        "exemplarCity";
        "مونروفيا";
    }

```



```

    }
    "Nairobi";
    {
        "exemplarCity";
        "نيروبي";
    }
    "Ndjamena";
    {
        "exemplarCity";
        "نجامينا";
    }
    "Niamey";
    {
        "exemplarCity";
        "نيامي";
    }
    "Nouakchott";
    {
        "exemplarCity";
        "نواكشوط";
    }
    "Ouagadougou";
    {
        "exemplarCity";
        "واغادوغو";
    }
    "Porto-Novo";
    {
        "exemplarCity";
        "بورتو نوفو";
    }
    "Sao_Tome";
    {
        "exemplarCity";
        "ساو تومي";
    }
    "Tripoli";
    {
        "exemplarCity";
        "طرابلس";
    }
    "Tunis";
    {
        "exemplarCity";
        "تونس";
    }
    "Windhoek";
    {
        "exemplarCity";
        "ويندهوك";
    }
}
"Asia";
{
    "Aden";
    {
        "exemplarCity";
    }
}

```

```
        "عدن";
    }
    "Almaty";
    {
        "exemplarCity";
        "ألماتي";
    }
    "Amman";
    {
        "exemplarCity";
        "عمان";
    }
    "Anadyr";
    {
        "exemplarCity";
        "أندير";
    }
    "Aqtan";
    {
        "exemplarCity";
        "أكتان";
    }
    "Aqtobe";
    {
        "exemplarCity";
        "أكتوب";
    }
    "Ashgabat";
    {
        "exemplarCity";
        "عشق آباد";
    }
    "Baghdad";
    {
        "exemplarCity";
        "بغداد";
    }
    "Bahrain";
    {
        "exemplarCity";
        "البحرين";
    }
    "Baku";
    {
        "exemplarCity";
        "باكو";
    }
    "Bangkok";
    {
        "exemplarCity";
        "بانكوك";
    }
    "Barnaul";
    {
        "exemplarCity";
        "بارناول";
    }
}
```

```
"Beirut";
{
  "exemplarCity";
  "بيروت";
}
"Bishkek";
{
  "exemplarCity";
  "بشكيك";
}
"Brunei";
{
  "exemplarCity";
  "بروناي";
}
"Calcutta";
{
  "exemplarCity";
  "كالكتا";
}
"Chita";
{
  "exemplarCity";
  "تشيتا";
}
"Choibalsan";
{
  "exemplarCity";
  "تشوبالسان";
}
"Colombo";
{
  "exemplarCity";
  "كولومبو";
}
"Damascus";
{
  "exemplarCity";
  "دمشق";
}
"Dhaka";
{
  "exemplarCity";
  "دكا";
}
"Dili";
{
  "exemplarCity";
  "ديلي";
}
"Dubai";
{
  "exemplarCity";
  "دبي";
}
"Dushanbe";
{
```

```

        "exemplarCity";
        "دوشانبي";
    }
    "Gaza";
    {
        "exemplarCity";
        "غزة";
    }
    "Hebron";
    {
        "exemplarCity";
        "هيبرون (مدينة الخليل)";
    }
    "Hong_Kong";
    {
        "exemplarCity";
        "هونغ كونغ";
    }
    "Hovd";
    {
        "exemplarCity";
        "هوفد";
    }
    "Irkutsk";
    {
        "exemplarCity";
        "ايركيتسك";
    }
    "Jakarta";
    {
        "exemplarCity";
        "جاكرتا";
    }
    "Jayapura";
    {
        "exemplarCity";
        "جاياپورا";
    }
    "Jerusalem";
    {
        "exemplarCity";
        "القدس";
    }
    "Kabul";
    {
        "exemplarCity";
        "كابول";
    }
    "Kamchatka";
    {
        "exemplarCity";
        "كامتشاتكا";
    }
    "Karachi";
    {
        "exemplarCity";
        "كراتشي";
    }

```

```
}
"Katmandu";
{
  "exemplarCity";
  "كاتماندو";
}
"Khandyga";
{
  "exemplarCity";
  "خاندیجا";
}
"Krasnoyarsk";
{
  "exemplarCity";
  "کراسنویا رسک";
}
"Kuala_Lumpur";
{
  "exemplarCity";
  "کوالا لامبور";
}
"Kuching";
{
  "exemplarCity";
  "کیشینج";
}
"Kuwait";
{
  "exemplarCity";
  "الکویت";
}
"Macau";
{
  "exemplarCity";
  "ماکاو";
}
"Magadan";
{
  "exemplarCity";
  "مجادن";
}
"Makassar";
{
  "exemplarCity";
  "ماکسار";
}
"Manila";
{
  "exemplarCity";
  "مانیلا";
}
"Muscat";
{
  "exemplarCity";
  "مسقط";
}
"Nicosia";
```

```

{
    "exemplarCity";
    "نيقوسيا";
}
"Novokuznetsk";
{
    "exemplarCity";
    "نوفوكوزنتسك";
}
"Novosibirsk";
{
    "exemplarCity";
    "نوفوسبيرسك";
}
"Omsk";
{
    "exemplarCity";
    "أومسك";
}
"Oral";
{
    "exemplarCity";
    "أورال";
}
"Phnom_Penh";
{
    "exemplarCity";
    "بنوم بنه";
}
"Pontianak";
{
    "exemplarCity";
    "بونتيانك";
}
"Pyongyang";
{
    "exemplarCity";
    "بيونغ يانغ";
}
"Qatar";
{
    "exemplarCity";
    "قطر";
}
"Qyzylorda";
{
    "exemplarCity";
    "كيزيلوردا";
}
"Rangoon";
{
    "exemplarCity";
    "رانغون";
}
"Riyadh";
{
    "exemplarCity";

```

```
        "الرياض";
    }
    "Saigon";
    {
        "exemplarCity";
        "مدينة هو تشي منه";
    }
    "Sakhalin";
    {
        "exemplarCity";
        "سكالين";
    }
    "Samarkand";
    {
        "exemplarCity";
        "سمرقند";
    }
    "Seoul";
    {
        "exemplarCity";
        "سول";
    }
    "Shanghai";
    {
        "exemplarCity";
        "شنغهاي";
    }
    "Singapore";
    {
        "exemplarCity";
        "سنغافورة";
    }
    "Srednekolymsk";
    {
        "exemplarCity";
        "سريدنكوليمسك";
    }
    "Taipei";
    {
        "exemplarCity";
        "تايبيه";
    }
    "Tashkent";
    {
        "exemplarCity";
        "تشقند";
    }
    "Tbilisi";
    {
        "exemplarCity";
        "تبليسي";
    }
    "Tehran";
    {
        "exemplarCity";
        "تهران";
    }
}
```

```
"Thimphu";
{
  "exemplarCity";
  "تیمفو";
}
"Tokyo";
{
  "exemplarCity";
  "طوكيو";
}
"Tomsk";
{
  "exemplarCity";
  "تومسك";
}
"Ulaanbaatar";
{
  "exemplarCity";
  "آلانباتار";
}
"Urumqi";
{
  "exemplarCity";
  "أرومقي";
}
"Ust-Nera";
{
  "exemplarCity";
  "أوست نيرا";
}
"Vientiane";
{
  "exemplarCity";
  "فيانتيان";
}
"Vladivostok";
{
  "exemplarCity";
  "فلاديفوستك";
}
"Yakutsk";
{
  "exemplarCity";
  "ياكتسك";
}
"Yekaterinburg";
{
  "exemplarCity";
  "يكاترنبيرج";
}
"Yerevan";
{
  "exemplarCity";
  "يريفان";
}
}
"Indian";
```



```

    {
      "Antananarivo";
      {
        "exemplarCity";
        "أنتاناناريفو";
      }
      "Chagos";
      {
        "exemplarCity";
        "تشاغوس";
      }
      "Christmas";
      {
        "exemplarCity";
        "كريسماس";
      }
      "Cocos";
      {
        "exemplarCity";
        "كوكوس";
      }
      "Comoro";
      {
        "exemplarCity";
        "جزر القمر";
      }
      "Kerguelen";
      {
        "exemplarCity";
        "كيرغويلين";
      }
      "Mahe";
      {
        "exemplarCity";
        "ماهي";
      }
      "Maldives";
      {
        "exemplarCity";
        "المالديف";
      }
      "Mauritius";
      {
        "exemplarCity";
        "موريشيوس";
      }
      "Mayotte";
      {
        "exemplarCity";
        "مايوت";
      }
      "Reunion";
      {
        "exemplarCity";
        "ريونيون";
      }
    }
  }

```

```
"Australia";
{
  "Adelaide";
  {
    "exemplarCity";
    "أديليد";
  }
  "Brisbane";
  {
    "exemplarCity";
    "برسبان";
  }
  "Broken_Hill";
  {
    "exemplarCity";
    "بروكن هيل";
  }
  "Currie";
  {
    "exemplarCity";
    "كوري";
  }
  "Darwin";
  {
    "exemplarCity";
    "دارون";
  }
  "Eucla";
  {
    "exemplarCity";
    "أوكلا";
  }
  "Hobart";
  {
    "exemplarCity";
    "هوبارت";
  }
  "Lindeman";
  {
    "exemplarCity";
    "ليندمان";
  }
  "Lord_Howe";
  {
    "exemplarCity";
    "لورد هاو";
  }
  "Melbourne";
  {
    "exemplarCity";
    "ميلبورن";
  }
  "Perth";
  {
    "exemplarCity";
    "برثا";
  }
}
```

```
"Sydney";
{
  "exemplarCity";
  "سيدني";
}
}
"Pacific";
{
  "Apia";
  {
    "exemplarCity";
    "أبيا";
  }
  "Auckland";
  {
    "exemplarCity";
    "أوكلاند";
  }
  "Bougainville";
  {
    "exemplarCity";
    "بوغانفيل";
  }
  "Chatham";
  {
    "exemplarCity";
    "تشاثام";
  }
  "Easter";
  {
    "exemplarCity";
    "استر";
  }
  "Efate";
  {
    "exemplarCity";
    "إيفات";
  }
  "Enderbury";
  {
    "exemplarCity";
    "اندربيرج";
  }
  "Fakaofu";
  {
    "exemplarCity";
    "فاكاوفو";
  }
  "Fiji";
  {
    "exemplarCity";
    "فيجي";
  }
  "Funafuti";
  {
    "exemplarCity";
    "فونافوتي";
  }
```

```
}
"Galapagos";
{
  "exemplarCity";
  "جلاپاجوس";
}
"Gambier";
{
  "exemplarCity";
  "جامبير";
}
"Guadalcanal";
{
  "exemplarCity";
  "غوادالكانال";
}
"Guam";
{
  "exemplarCity";
  "غوام";
}
"Honolulu";
{
  "exemplarCity";
  "هونولولو";
}
"Johnston";
{
  "exemplarCity";
  "جونستون";
}
"Kiritimati";
{
  "exemplarCity";
  "كيريتي ماتي";
}
"Kosrae";
{
  "exemplarCity";
  "كوسرا";
}
"Kwajalein";
{
  "exemplarCity";
  "كوجالين";
}
"Majuro";
{
  "exemplarCity";
  "ماجورو";
}
"Marquesas";
{
  "exemplarCity";
  "ماركيساس";
}
"Midway";
```

```
{
    "exemplarCity";
    "ميدواي";
}
"Nauru";
{
    "exemplarCity";
    "ناورو";
}
"Niue";
{
    "exemplarCity";
    "نيوي";
}
"Norfolk";
{
    "exemplarCity";
    "نورفولك";
}
"Noumea";
{
    "exemplarCity";
    "نوميا";
}
"Pago_Pago";
{
    "exemplarCity";
    "باغو باغو";
}
"Palau";
{
    "exemplarCity";
    "بالاو";
}
"Pitcairn";
{
    "exemplarCity";
    "بيتكيرن";
}
"Ponape";
{
    "exemplarCity";
    "باناب";
}
"Port_Moresby";
{
    "exemplarCity";
    "بور مورسبي";
}
"Rarotonga";
{
    "exemplarCity";
    "راروتونغا";
}
"Saipan";
{
    "exemplarCity";
```

```

        "سايبان";
    }
    "Tahiti";
    {
        "exemplarCity";
        "تاهيتي";
    }
    "Tarawa";
    {
        "exemplarCity";
        "تاراوا";
    }
    "Tongatapu";
    {
        "exemplarCity";
        "تونغاتابو";
    }
    "Truk";
    {
        "exemplarCity";
        "ترك";
    }
    "Wake";
    {
        "exemplarCity";
        "واك";
    }
    "Wallis";
    {
        "exemplarCity";
        "واليس";
    }
}
"Arctic";
{
    "Longyearbyen";
    {
        "exemplarCity";
        "لونغجيربين";
    }
}
"Antarctica";
{
    "Casey";
    {
        "exemplarCity";
        "كاساي";
    }
    "Davis";
    {
        "exemplarCity";
        "دافيز";
    }
    "DumontDUrville";
    {
        "exemplarCity";
        "دي مونت دو روفيل";
    }
}

```

```
}
"Macquarie";
{
  "exemplarCity";
  "ماكواري";
}
"Mawson";
{
  "exemplarCity";
  "ماوسون";
}
"McMurdo";
{
  "exemplarCity";
  "ماك موربدو";
}
"Palmer";
{
  "exemplarCity";
  "بالمير";
}
"Rothera";
{
  "exemplarCity";
  "روثيرا";
}
"Syowa";
{
  "exemplarCity";
  "سايووا";
}
"Troll";
{
  "exemplarCity";
  "ترول";
}
"Vostok";
{
  "exemplarCity";
  "فوستوك";
}
}
"Etc";
{
  "GMT";
  {
    "exemplarCity";
    "GMT";
  }
  "GMT1";
  {
    "exemplarCity";
    "GMT+1";
  }
  "GMT10";
  {
    "exemplarCity";
```

```
        "GMT+10";
    }
    "GMT11";
    {
        "exemplarCity";
        "GMT+11";
    }
    "GMT12";
    {
        "exemplarCity";
        "GMT+12";
    }
    "GMT2";
    {
        "exemplarCity";
        "GMT+2";
    }
    "GMT3";
    {
        "exemplarCity";
        "GMT+3";
    }
    "GMT4";
    {
        "exemplarCity";
        "GMT+4";
    }
    "GMT5";
    {
        "exemplarCity";
        "GMT+5";
    }
    "GMT6";
    {
        "exemplarCity";
        "GMT+6";
    }
    "GMT7";
    {
        "exemplarCity";
        "GMT+7";
    }
    "GMT8";
    {
        "exemplarCity";
        "GMT+8";
    }
    "GMT9";
    {
        "exemplarCity";
        "GMT+9";
    }
    "GMT-1";
    {
        "exemplarCity";
        "GMT-1";
    }
}
```



```
"GMT-10";
{
  "exemplarCity";
  "GMT-10";
}
"GMT-11";
{
  "exemplarCity";
  "GMT-11";
}
"GMT-12";
{
  "exemplarCity";
  "GMT-12";
}
"GMT-13";
{
  "exemplarCity";
  "GMT-13";
}
"GMT-14";
{
  "exemplarCity";
  "GMT-14";
}
"GMT-2";
{
  "exemplarCity";
  "GMT-2";
}
"GMT-3";
{
  "exemplarCity";
  "GMT-3";
}
"GMT-4";
{
  "exemplarCity";
  "GMT-4";
}
"GMT-5";
{
  "exemplarCity";
  "GMT-5";
}
"GMT-6";
{
  "exemplarCity";
  "GMT-6";
}
"GMT-7";
{
  "exemplarCity";
  "GMT-7";
}
"GMT-8";
{
```

```

        "exemplarCity";
        "GMT-8";
    }
    "GMT-9";
    {
        "exemplarCity";
        "GMT-9";
    }
    "Unknown";
    {
        "exemplarCity";
        "مدينة غير معروفة";
    }
}
"metazone";
{
    "Afghanistan";
    {
        "long";
        {
            "standard";
            "توقيت أفغانستان";
        }
    }
    "Africa_Central";
    {
        "long";
        {
            "standard";
            "توقيت وسط أفريقيا";
        }
    }
    "Africa_Eastern";
    {
        "long";
        {
            "standard";
            "توقيت شرق أفريقيا";
        }
    }
    "Africa_Southern";
    {
        "long";
        {
            "standard";
            "توقيت جنوب أفريقيا";
        }
    }
    "Africa_Western";
    {
        "long";
        {
            "generic";
            "توقيت غرب أفريقيا",
            "standard";
            "توقيت غرب أفريقيا الرسمي",

```

```

        "daylight";
        "توقيت غرب أفريقيا الصيفي";
    }
}
"Alaska";
{
    "long";
    {
        "generic";
        "توقيت ألاسكا",
        "standard";
        "التوقيت الرسمي لألاسكا",
        "daylight";
        "توقيت ألاسكا الصيفي";
    }
}
"Amazon";
{
    "long";
    {
        "generic";
        "توقيت الأمازون",
        "standard";
        "توقيت الأمازون الرسمي",
        "daylight";
        "توقيت الأمازون الصيفي";
    }
}
"America_Central";
{
    "long";
    {
        "generic";
        "التوقيت المركزي لأمريكا الشمالية",
        "standard";
        "التوقيت الرسمي المركزي لأمريكا الشمالية",
        "daylight";
        "التوقيت الصيفي المركزي لأمريكا الشمالية";
    }
}
"America_Eastern";
{
    "long";
    {
        "generic";
        "التوقيت الشرقي لأمريكا الشمالية",
        "standard";
        "التوقيت الرسمي الشرقي لأمريكا الشمالية",
        "daylight";
        "التوقيت الصيفي الشرقي لأمريكا الشمالية";
    }
}
"America_Mountain";
{
    "long";
    {
        "generic";

```

```

        "التوقيت الجبلي لأمريكا الشمالية",
        "standard";
        "التوقيت الجبلي الرسمي لأمريكا الشمالية",
        "daylight";
        "التوقيت الجبلي الصيفي لأمريكا الشمالية";
    }
}
"America_Pacific";
{
    "long";
    {
        "generic";
        "توقيت المحيط الهادي",
        "standard";
        "توقيت المحيط الهادي الرسمي",
        "daylight";
        "توقيت المحيط الهادي الصيفي";
    }
}
"Anadyr";
{
    "long";
    {
        "generic";
        "توقيت أنادير",
        "standard";
        "توقيت أنادير الرسمي",
        "daylight";
        "التوقيت الصيفي لأنادير";
    }
}
"Apia";
{
    "long";
    {
        "generic";
        "توقيت آبيا",
        "standard";
        "التوقيت الرسمي لآبيا",
        "daylight";
        "التوقيت الصيفي لآبيا";
    }
}
"Arabian";
{
    "long";
    {
        "generic";
        "التوقيت العربي",
        "standard";
        "التوقيت العربي الرسمي",
        "daylight";
        "التوقيت العربي الصيفي";
    }
}
"Argentina";
{

```

```

        "long";
        {
            "generic";
            "توقيت الأرجنتين",
            "standard";
            "توقيت الأرجنتين الرسمي",
            "daylight";
            "توقيت الأرجنتين الصيفي";
        }
    }
    "Argentina_Western";
    {
        "long";
        {
            "generic";
            "توقيت غرب الأرجنتين",
            "standard";
            "توقيت غرب الأرجنتين الرسمي",
            "daylight";
            "توقيت غرب الأرجنتين الصيفي";
        }
    }
    "Armenia";
    {
        "long";
        {
            "generic";
            "توقيت أرمينيا",
            "standard";
            "توقيت أرمينيا الرسمي",
            "daylight";
            "توقيت أرمينيا الصيفي";
        }
    }
    "Atlantic";
    {
        "long";
        {
            "generic";
            "توقيت الأطلسي",
            "standard";
            "التوقيت الرسمي الأطلسي",
            "daylight";
            "التوقيت الصيفي الأطلسي";
        }
    }
    "Australia_Central";
    {
        "long";
        {
            "generic";
            "توقيت وسط أستراليا",
            "standard";
            "توقيت وسط أستراليا الرسمي",
            "daylight";
            "توقيت وسط أستراليا الصيفي";
        }
    }

```

```

    }
    "Australia_CentralWestern";
    {
        "long";
        {
            "generic";
            "توقيت غرب وسط أستراليا",
            "standard";
            "توقيت غرب وسط أستراليا الرسمي",
            "daylight";
            "توقيت غرب وسط أستراليا الصيفي";
        }
    }
    "Australia_Eastern";
    {
        "long";
        {
            "generic";
            "توقيت شرق أستراليا",
            "standard";
            "توقيت شرق أستراليا الرسمي",
            "daylight";
            "توقيت شرق أستراليا الصيفي";
        }
    }
    "Australia_Western";
    {
        "long";
        {
            "generic";
            "توقيت غرب أستراليا",
            "standard";
            "توقيت غرب أستراليا الرسمي",
            "daylight";
            "توقيت غرب أستراليا الصيفي";
        }
    }
    "Azerbaijan";
    {
        "long";
        {
            "generic";
            "توقيت أذربيجان",
            "standard";
            "توقيت أذربيجان الرسمي",
            "daylight";
            "توقيت أذربيجان الصيفي";
        }
    }
    "Azores";
    {
        "long";
        {
            "generic";
            "توقيت أزورس",
            "standard";
            "توقيت أزورس الرسمي",

```

```

        "daylight";
        "توقيت أزورس الصيفي";
    }
}
"Bangladesh";
{
    "long";
    {
        "generic";
        "توقيت بنجلاديش",
        "standard";
        "توقيت بنجلاديش الرسمي",
        "daylight";
        "توقيت بنجلاديش الصيفي";
    }
}
"Bhutan";
{
    "long";
    {
        "standard";
        "توقيت بوتان";
    }
}
"Bolivia";
{
    "long";
    {
        "standard";
        "توقيت بوليفيا";
    }
}
"Brasilia";
{
    "long";
    {
        "generic";
        "توقيت برازيليا",
        "standard";
        "توقيت برازيليا الرسمي",
        "daylight";
        "توقيت برازيليا الصيفي";
    }
}
"Brunei";
{
    "long";
    {
        "standard";
        "توقيت بروناي";
    }
}
"Cape_Verde";
{
    "long";
    {
        "generic";

```

```

        "توقيت الرأس الأخضر",
        "standard";
        "توقيت الرأس الأخضر الرسمي",
        "daylight";
        "توقيت الرأس الأخضر الصيفي";
    }
}
"Chamorro";
{
    "long";
    {
        "standard";
        "توقيت تشامورو";
    }
}
"Chatham";
{
    "long";
    {
        "generic";
        "توقيت تشاتام",
        "standard";
        "توقيت تشاتام الرسمي",
        "daylight";
        "توقيت تشاتام الصيفي";
    }
}
"Chile";
{
    "long";
    {
        "generic";
        "توقيت شيلي",
        "standard";
        "توقيت شيلي الرسمي",
        "daylight";
        "توقيت شيلي الصيفي";
    }
}
"China";
{
    "long";
    {
        "generic";
        "توقيت الصين",
        "standard";
        "توقيت الصين الرسمي",
        "daylight";
        "توقيت الصين الصيفي";
    }
}
"Choibalsan";
{
    "long";
    {
        "generic";
        "توقيت شويبالسان",

```



```

        "standard";
        "توقيت شويبالسان الرسمي",
        "daylight";
        "التوقيت الصيفي لشويبالسان";
    }
}
"Christmas";
{
    "long";
    {
        "standard";
        "توقيت جزر الكريسماس";
    }
}
"Cocos";
{
    "long";
    {
        "standard";
        "توقيت جزر كوكوس";
    }
}
"Colombia";
{
    "long";
    {
        "generic";
        "توقيت كولومبيا",
        "standard";
        "توقيت كولومبيا الرسمي",
        "daylight";
        "توقيت كولومبيا الصيفي";
    }
}
"Cook";
{
    "long";
    {
        "generic";
        "توقيت جزر كوك",
        "standard";
        "توقيت جزر كوك الرسمي",
        "daylight";
        "توقيت جزر كوك الصيفي";
    }
}
"Cuba";
{
    "long";
    {
        "generic";
        "توقيت كوبا",
        "standard";
        "توقيت كوبا الرسمي",
        "daylight";
        "توقيت كوبا الصيفي";
    }
}

```

```

    }
    "Davis";
    {
        "long";
        {
            "standard";
            "توقيت دافيز";
        }
    }
    "DumontDUrville";
    {
        "long";
        {
            "standard";
            "توقيت دي مونت دو روفيل";
        }
    }
    "East_Timor";
    {
        "long";
        {
            "standard";
            "توقيت تيمور الشرقية";
        }
    }
    "Easter";
    {
        "long";
        {
            "generic";
            "توقيت جزيرة استر",
            "standard";
            "توقيت جزيرة استر الرسمي",
            "daylight";
            "توقيت جزيرة استر الصيفي";
        }
    }
    "Ecuador";
    {
        "long";
        {
            "standard";
            "توقيت الإكوادور";
        }
    }
    "Europe_Central";
    {
        "long";
        {
            "generic";
            "توقيت وسط أوروبا",
            "standard";
            "توقيت وسط أوروبا الرسمي",
            "daylight";
            "توقيت وسط أوروبا الصيفي";
        }
    }
}

```

```

"Europe_Eastern";
{
  "long";
  {
    "generic";
    "توقيت شرق أوروبا",
    "standard";
    "توقيت شرق أوروبا الرسمي",
    "daylight";
    "توقيت شرق أوروبا الصيفي";
  }
}
"Europe_Further_Eastern";
{
  "long";
  {
    "standard";
    "التوقيت الأوروبي (أكثر شرقًا)";
  }
}
"Europe_Western";
{
  "long";
  {
    "generic";
    "توقيت غرب أوروبا",
    "standard";
    "توقيت غرب أوروبا الرسمي",
    "daylight";
    "توقيت غرب أوروبا الصيفي";
  }
}
"Falkland";
{
  "long";
  {
    "generic";
    "توقيت جزر فوكلاند",
    "standard";
    "توقيت جزر فوكلاند الرسمي",
    "daylight";
    "توقيت جزر فوكلاند الصيفي";
  }
}
"Fiji";
{
  "long";
  {
    "generic";
    "توقيت فيجي",
    "standard";
    "توقيت فيجي الرسمي",
    "daylight";
    "توقيت فيجي الصيفي";
  }
}
"French_Guiana";

```

```

        {
            "long";
            {
                "standard";
                "توقيت غايانا الفرنسية";
            }
        }
        "French_Southern";
        {
            "long";
            {
                "standard";
                "توقيت المقاطعات الفرنسية الجنوبية";
            }
        }
        "Galapagos";
        {
            "long";
            {
                "standard";
                "توقيت غلاباغوس";
            }
        }
        "Gambier";
        {
            "long";
            {
                "standard";
                "توقيت جامبير";
            }
        }
        "Georgia";
        {
            "long";
            {
                "generic";
                "توقيت جورجيا",
                "standard";
                "توقيت جورجيا الرسمي",
                "daylight";
                "توقيت جورجيا الصيفي";
            }
        }
        "Gilbert_Islands";
        {
            "long";
            {
                "standard";
                "توقيت جزر جيلبرت";
            }
        }
        "GMT";
        {
            "long";
            {
                "standard";
            }
        }
    }

```

```

        "توقيت غرينتش";
    }
}
"Greenland_Eastern";
{
    "long";
    {
        "generic";
        "توقيت شرق غرينلاند",
        "standard";
        "توقيت شرق غرينلاند الرسمي",
        "daylight";
        "توقيت شرق غرينلاند الصيفي";
    }
}
"Greenland_Western";
{
    "long";
    {
        "generic";
        "توقيت غرب غرينلاند",
        "standard";
        "توقيت غرب غرينلاند الرسمي",
        "daylight";
        "توقيت غرب غرينلاند الصيفي";
    }
}
"Guam";
{
    "long";
    {
        "standard";
        "توقيت غوام";
    }
}
"Gulf";
{
    "long";
    {
        "standard";
        "توقيت الخليج";
    }
}
"Guyana";
{
    "long";
    {
        "standard";
        "توقيت غيانا";
    }
}
"Hawaii_Aleutian";
{
    "long";
    {
        "generic";
        "توقيت هاواي ألوتيان",

```

```

        "standard";
        "توقيت هاواي ألتيان الرسمي",
        "daylight";
        "توقيت هاواي ألتيان الصيفي";
    }
}
"Hong_Kong";
{
    "long";
    {
        "generic";
        "توقيت هونغ كونغ",
        "standard";
        "توقيت هونغ كونغ الرسمي",
        "daylight";
        "توقيت هونغ كونغ الصيفي";
    }
}
"Hovd";
{
    "long";
    {
        "generic";
        "توقيت هوفد",
        "standard";
        "توقيت هوفد الرسمي",
        "daylight";
        "توقيت هوفد الصيفي";
    }
}
"India";
{
    "long";
    {
        "standard";
        "توقيت الهند";
    }
}
"Indian_Ocean";
{
    "long";
    {
        "standard";
        "توقيت المحيط الهندي";
    }
}
"Indochina";
{
    "long";
    {
        "standard";
        "توقيت الهند الصينية";
    }
}
"Indonesia_Central";
{
    "long";

```

```

        {
            "standard";
            "توقيت وسط إندونيسيا";
        }
    }
    "Indonesia_Eastern";
    {
        "long";
        {
            "standard";
            "توقيت شرق إندونيسيا";
        }
    }
    "Indonesia_Western";
    {
        "long";
        {
            "standard";
            "توقيت غرب إندونيسيا";
        }
    }
    "Iran";
    {
        "long";
        {
            "generic";
            "توقيت إيران",
            "standard";
            "توقيت إيران الرسمي",
            "daylight";
            "توقيت إيران الصيفي";
        }
    }
    "Irkutsk";
    {
        "long";
        {
            "generic";
            "توقيت إركوتسك",
            "standard";
            "توقيت إركوتسك الرسمي",
            "daylight";
            "توقيت إركوتسك الصيفي";
        }
    }
    "Israel";
    {
        "long";
        {
            "generic";
            "توقيت إسرائيل",
            "standard";
            "توقيت إسرائيل الرسمي",
            "daylight";
            "توقيت إسرائيل الصيفي";
        }
    }
}

```

```

"Japan";
{
  "long";
  {
    "generic";
    "توقيت اليابان",
    "standard";
    "توقيت اليابان الرسمي",
    "daylight";
    "توقيت اليابان الصيفي";
  }
}
"Kamchatka";
{
  "long";
  {
    "generic";
    "توقيت كامشاتكا",
    "standard";
    "توقيت بيتروبافلوفسك-كامتشاتسكي",
    "daylight";
    "توقيت بيتروبافلوفسك-كامتشاتسكي الصيفي";
  }
}
"Kazakhstan_Eastern";
{
  "long";
  {
    "standard";
    "توقيت شرق كازاخستان";
  }
}
"Kazakhstan_Western";
{
  "long";
  {
    "standard";
    "توقيت غرب كازاخستان";
  }
}
"Korea";
{
  "long";
  {
    "generic";
    "توقيت كوريا",
    "standard";
    "توقيت كوريا الرسمي",
    "daylight";
    "توقيت كوريا الصيفي";
  }
}
"Kosrae";
{
  "long";
  {
    "standard";

```



```

        "توقيت كوسرا";
    }
}
"Krasnoyarsk";
{
    "long";
    {
        "generic";
        "توقيت كراسنويارسك",
        "standard";
        "توقيت كراسنويارسك الرسمي",
        "daylight";
        "التوقيت الصيفي لكراسنويارسك";
    }
}
"Kyrgystan";
{
    "long";
    {
        "standard";
        "توقيت قرغيزستان";
    }
}
"Line_Islands";
{
    "long";
    {
        "standard";
        "توقيت جزر لاين";
    }
}
"Lord_Howe";
{
    "long";
    {
        "generic";
        "توقيت لورد هاو",
        "standard";
        "توقيت لورد هاو الرسمي",
        "daylight";
        "التوقيت الصيفي للورد هاو";
    }
}
"Macquarie";
{
    "long";
    {
        "standard";
        "توقيت ماكوارى";
    }
}
"Magadan";
{
    "long";
    {
        "generic";
        "توقيت ماغادان",

```

```

        "standard";
        "توقيت ماغادان الرسمي",
        "daylight";
        "توقيت ماغادان الصيفي";
    }
}
"Malaysia";
{
    "long";
    {
        "standard";
        "توقيت ماليزيا";
    }
}
"Maldives";
{
    "long";
    {
        "standard";
        "توقيت المالديف";
    }
}
"Marquesas";
{
    "long";
    {
        "standard";
        "توقيت ماركيساس";
    }
}
"Marshall_Islands";
{
    "long";
    {
        "standard";
        "توقيت جزر مارشال";
    }
}
"Mauritius";
{
    "long";
    {
        "generic";
        "توقيت موريشيوس",
        "standard";
        "توقيت موريشيوس الرسمي",
        "daylight";
        "توقيت موريشيوس الصيفي";
    }
}
"Mawson";
{
    "long";
    {
        "standard";
        "توقيت ماوسون";
    }
}

```

```

    }
    "Mexico_Northwest";
    {
        "long";
        {
            "generic";
            "توقيت شمال غرب المكسيك",
            "standard";
            "التوقيت الرسمي لشمال غرب المكسيك",
            "daylight";
            "التوقيت الصيفي لشمال غرب المكسيك";
        }
    }
    "Mexico_Pacific";
    {
        "long";
        {
            "generic";
            "توقيت المحيط الهادي للمكسيك",
            "standard";
            "توقيت المحيط الهادي الرسمي للمكسيك",
            "daylight";
            "توقيت المحيط الهادي الصيفي للمكسيك";
        }
    }
    "Mongolia";
    {
        "long";
        {
            "generic";
            "توقيت أولان باتور",
            "standard";
            "توقيت أولان باتور الرسمي",
            "daylight";
            "توقيت أولان باتور الصيفي";
        }
    }
    "Moscow";
    {
        "long";
        {
            "generic";
            "توقيت موسكو",
            "standard";
            "توقيت موسكو الرسمي",
            "daylight";
            "توقيت موسكو الصيفي";
        }
    }
    "Myanmar";
    {
        "long";
        {
            "standard";
            "توقيت ميانمار";
        }
    }
}

```

```

"Nauru";
{
  "long";
  {
    "standard";
    "توقيت ناورو";
  }
}
"Nepal";
{
  "long";
  {
    "standard";
    "توقيت نيبال";
  }
}
"New_Caledonia";
{
  "long";
  {
    "generic";
    "توقيت كاليدونيا الجديدة",
    "standard";
    "توقيت كاليدونيا الجديدة الرسمي",
    "daylight";
    "توقيت كاليدونيا الجديدة الصيفي";
  }
}
"New_Zealand";
{
  "long";
  {
    "generic";
    "توقيت نيوزيلندا",
    "standard";
    "توقيت نيوزيلندا الرسمي",
    "daylight";
    "توقيت نيوزيلندا الصيفي";
  }
}
"Newfoundland";
{
  "long";
  {
    "generic";
    "توقيت نيوفاوندلاند",
    "standard";
    "توقيت نيوفاوندلاند الرسمي",
    "daylight";
    "توقيت نيوفاوندلاند الصيفي";
  }
}
"Niue";
{
  "long";
  {
    "standard";

```

```

        "توقيت نيوي";
    }
}
"Norfolk";
{
    "long";
    {
        "standard";
        "توقيت جزيرة نورفولك";
    }
}
"Noronha";
{
    "long";
    {
        "generic";
        "توقيت فيرناندو دي نورونها",
        "standard";
        "توقيت فرناندو دي نورونها الرسمي",
        "daylight";
        "توقيت فرناندو دي نورونها الصيفي";
    }
}
"North_Mariana";
{
    "long";
    {
        "standard";
        "توقيت جزر ماريانا الشمالية";
    }
}
"Novosibirsk";
{
    "long";
    {
        "generic";
        "توقيت نوفوسيبيرسك",
        "standard";
        "توقيت نوفوسيبيرسك الرسمي",
        "daylight";
        "توقيت نوفوسيبيرسك الصيفي";
    }
}
"Omsk";
{
    "long";
    {
        "generic";
        "توقيت أومسك",
        "standard";
        "توقيت أومسك الرسمي",
        "daylight";
        "توقيت أومسك الصيفي";
    }
}
"Pakistan";
{

```

```

        "long";
        {
            "generic";
            "توقيت باكستان",
            "standard";
            "توقيت باكستان الرسمي",
            "daylight";
            "توقيت باكستان الصيفي";
        }
    }
    "Palau";
    {
        "long";
        {
            "standard";
            "توقيت بالاو";
        }
    }
    "Papua_New_Guinea";
    {
        "long";
        {
            "standard";
            "توقيت بابوا غينيا الجديدة";
        }
    }
    "Paraguay";
    {
        "long";
        {
            "generic";
            "توقيت باراغواي",
            "standard";
            "توقيت باراغواي الرسمي",
            "daylight";
            "توقيت باراغواي الصيفي";
        }
    }
    "Peru";
    {
        "long";
        {
            "generic";
            "توقيت بيرو",
            "standard";
            "توقيت بيرو الرسمي",
            "daylight";
            "توقيت بيرو الصيفي";
        }
    }
    "Philippines";
    {
        "long";
        {
            "generic";
            "توقيت الفلبين",
            "standard";

```

```

        "توقيت الفيلبين الرسمي",
        "daylight";
        "توقيت الفيلبين الصيفي";
    }
}
"Phoenix_Islands";
{
    "long";
    {
        "standard";
        "توقيت جزر فينكس";
    }
}
"Pierre_Miquelon";
{
    "long";
    {
        "generic";
        "توقيت سانت بيير وميكلون",
        "standard";
        "توقيت سانت بيير وميكلون الرسمي",
        "daylight";
        "توقيت سانت بيير وميكلون الصيفي";
    }
}
"Pitcairn";
{
    "long";
    {
        "standard";
        "توقيت بيتكيرن";
    }
}
"Ponape";
{
    "long";
    {
        "standard";
        "توقيت بونابي";
    }
}
"Pyongyang";
{
    "long";
    {
        "standard";
        "توقيت بيونغ يانغ";
    }
}
"Reunion";
{
    "long";
    {
        "standard";
        "توقيت ريونيون";
    }
}
}

```

```

"Rothera";
{
  "long";
  {
    "standard";
    "توقيت روثيرا";
  }
}
"Sakhalin";
{
  "long";
  {
    "generic";
    "توقيت ساخالين",
    "standard";
    "توقيت ساخالين الرسمي",
    "daylight";
    "توقيت ساخالين الصيفي";
  }
}
"Samara";
{
  "long";
  {
    "generic";
    "توقيت سامارا",
    "standard";
    "توقيت سمارة",
    "daylight";
    "توقيت سمارة الصيفي";
  }
}
"Samoa";
{
  "long";
  {
    "generic";
    "توقيت ساموا",
    "standard";
    "توقيت ساموا الرسمي",
    "daylight";
    "توقيت ساموا الصيفي";
  }
}
"Seychelles";
{
  "long";
  {
    "standard";
    "توقيت سيشل";
  }
}
"Singapore";
{
  "long";
  {
    "standard";
  }
}

```



```

        "توقيت سنغافورة";
    }
}
"Solomon";
{
    "long";
    {
        "standard";
        "توقيت جزر سليمان";
    }
}
"South_Georgia";
{
    "long";
    {
        "standard";
        "توقيت جنوب جورجيا";
    }
}
"Suriname";
{
    "long";
    {
        "standard";
        "توقيت سورينام";
    }
}
"Syowa";
{
    "long";
    {
        "standard";
        "توقيت سايووا";
    }
}
"Tahiti";
{
    "long";
    {
        "standard";
        "توقيت تاهيتي";
    }
}
"Taipei";
{
    "long";
    {
        "generic";
        "توقيت تايبيه",
        "standard";
        "توقيت تايبيه الرسمي",
        "daylight";
        "توقيت تايبيه الصيفي";
    }
}
"Tajikistan";
{

```

```

        "long";
        {
            "standard";
            "توقيت طاجكستان";
        }
    }
    "Tokelau";
    {
        "long";
        {
            "standard";
            "توقيت توكيلاو";
        }
    }
    "Tonga";
    {
        "long";
        {
            "generic";
            "توقيت تونغا",
            "standard";
            "توقيت تونغا الرسمي",
            "daylight";
            "توقيت تونغا الصيفي";
        }
    }
    "Truk";
    {
        "long";
        {
            "standard";
            "توقيت شوك";
        }
    }
    "Turkmenistan";
    {
        "long";
        {
            "generic";
            "توقيت تركمانستان",
            "standard";
            "توقيت تركمانستان الرسمي",
            "daylight";
            "توقيت تركمانستان الصيفي";
        }
    }
    "Tuvalu";
    {
        "long";
        {
            "standard";
            "توقيت توفالو";
        }
    }
    "Uruguay";
    {
        "long";

```

```

        {
            "generic";
            "توقيت أورغواي",
            "standard";
            "توقيت أورغواي الرسمي",
            "daylight";
            "توقيت أورغواي الصيفي";
        }
    }
    "Uzbekistan";
    {
        "long";
        {
            "generic";
            "توقيت أوزبكستان",
            "standard";
            "توقيت أوزبكستان الرسمي",
            "daylight";
            "توقيت أوزبكستان الصيفي";
        }
    }
    "Vanuatu";
    {
        "long";
        {
            "generic";
            "توقيت فانواتو",
            "standard";
            "توقيت فانواتو الرسمي",
            "daylight";
            "توقيت فانواتو الصيفي";
        }
    }
    "Venezuela";
    {
        "long";
        {
            "standard";
            "توقيت فنزويلا";
        }
    }
    "Vladivostok";
    {
        "long";
        {
            "generic";
            "توقيت فلاديفوستوك",
            "standard";
            "توقيت فلاديفوستوك الرسمي",
            "daylight";
            "توقيت فلاديفوستوك الصيفي";
        }
    }
    "Volgograd";
    {
        "long";
        {

```

```

        "generic";
        "توقيت فولغوغراد",
        "standard";
        "توقيت فولغوغراد الرسمي",
        "daylight";
        "توقيت فولغوغراد الصيفي";
    }
}
"Vostok";
{
    "long";
    {
        "standard";
        "توقيت فوستوك";
    }
}
"Wake";
{
    "long";
    {
        "standard";
        "توقيت جزيرة ويك";
    }
}
"Wallis";
{
    "long";
    {
        "standard";
        "توقيت واليس و فوتونا";
    }
}
"Yakutsk";
{
    "long";
    {
        "generic";
        "توقيت ياكوتسك",
        "standard";
        "توقيت ياكوتسك الرسمي",
        "daylight";
        "توقيت ياكوتسك الصيفي";
    }
}
"Yekaterinburg";
{
    "long";
    {
        "generic";
        "توقيت يكاترينبورغ",
        "standard";
        "توقيت يكاترينبورغ الرسمي",
        "daylight";
        "توقيت يكاترينبورغ الصيفي";
    }
}
}

```

```

    }
  }
}
}
}

```

[Functional-component]

CA-GREGORIAN.JSON

```

{
  "main": {
    "ar": {
      "identity": {
        "version": {
          "_number": "$Revision: 12879 $",
          "_cldrVersion": "30.0.3"
        },
        "language": "ar"
      },
      "dates": {
        "calendars": {
          "gregorian": {
            "months": {
              "format": {
                "abbreviated": {
                  "1": "يناير",
                  "2": "فبراير",
                  "3": "مارس",
                  "4": "أبريل",
                  "5": "مايو",
                  "6": "يونيو",
                  "7": "يوليو",
                  "8": "أغسطس",
                  "9": "سبتمبر",
                  "10": "أكتوبر",
                  "11": "نوفمبر",
                  "12": "ديسمبر"
                },
                "narrow": {
                  "1": "ي",
                  "2": "ف",
                  "3": "م",
                  "4": "أ",
                  "5": "و",
                  "6": "ن",
                  "7": "ل",
                  "8": "غ",
                  "9": "س",
                  "10": "ك",
                  "11": "ب",
                  "12": "د"
                },
                "wide": {
                  "1": "يناير",
                  "2": "فبراير",

```

```

        "3": "مارس",
        "4": "أبريل",
        "5": "مايو",
        "6": "يونيو",
        "7": "يوليو",
        "8": "أغسطس",
        "9": "سبتمبر",
        "10": "أكتوبر",
        "11": "نوفمبر",
        "12": "ديسمبر"
    },
    },
    "stand-alone": {
        "abbreviated": {
            "1": "يناير",
            "2": "فبراير",
            "3": "مارس",
            "4": "أبريل",
            "5": "مايو",
            "6": "يونيو",
            "7": "يوليو",
            "8": "أغسطس",
            "9": "سبتمبر",
            "10": "أكتوبر",
            "11": "نوفمبر",
            "12": "ديسمبر"
        },
        "narrow": {
            "1": "ي",
            "2": "ف",
            "3": "م",
            "4": "أ",
            "5": "و",
            "6": "ن",
            "7": "ل",
            "8": "غ",
            "9": "س",
            "10": "ك",
            "11": "ب",
            "12": "د"
        },
        "wide": {
            "1": "يناير",
            "2": "فبراير",
            "3": "مارس",
            "4": "أبريل",
            "5": "مايو",
            "6": "يونيو",
            "7": "يوليو",
            "8": "أغسطس",
            "9": "سبتمبر",
            "10": "أكتوبر",
            "11": "نوفمبر",
            "12": "ديسمبر"
        }
    }
},
},

```

```

"days": {
  "format": {
    "abbreviated": {
      "sun": "الأحد",
      "mon": "الاثنين",
      "tue": "الثلاثاء",
      "wed": "الأربعاء",
      "thu": "الخميس",
      "fri": "الجمعة",
      "sat": "السبت"
    },
    "narrow": {
      "sun": "ح",
      "mon": "ن",
      "tue": "ث",
      "wed": "ر",
      "thu": "خ",
      "fri": "ج",
      "sat": "س"
    },
    "short": {
      "sun": "الأحد",
      "mon": "الاثنين",
      "tue": "الثلاثاء",
      "wed": "الأربعاء",
      "thu": "الخميس",
      "fri": "الجمعة",
      "sat": "السبت"
    },
    "wide": {
      "sun": "الأحد",
      "mon": "الاثنين",
      "tue": "الثلاثاء",
      "wed": "الأربعاء",
      "thu": "الخميس",
      "fri": "الجمعة",
      "sat": "السبت"
    }
  },
  "stand-alone": {
    "abbreviated": {
      "sun": "الأحد",
      "mon": "الاثنين",
      "tue": "الثلاثاء",
      "wed": "الأربعاء",
      "thu": "الخميس",
      "fri": "الجمعة",
      "sat": "السبت"
    },
    "narrow": {
      "sun": "ح",
      "mon": "ن",
      "tue": "ث",
      "wed": "ر",
      "thu": "خ",
      "fri": "ج",
      "sat": "س"
    }
  }
}

```

```

    },
    "short": {
      "sun": "الأحد",
      "mon": "الاثنين",
      "tue": "الثلاثاء",
      "wed": "الأربعاء",
      "thu": "الخميس",
      "fri": "الجمعة",
      "sat": "السبت"
    },
    "wide": {
      "sun": "الأحد",
      "mon": "الاثنين",
      "tue": "الثلاثاء",
      "wed": "الأربعاء",
      "thu": "الخميس",
      "fri": "الجمعة",
      "sat": "السبت"
    }
  },
  "quarters": {
    "format": {
      "abbreviated": {
        "1": "الربع الأول",
        "2": "الربع الثاني",
        "3": "الربع الثالث",
        "4": "الربع الرابع"
      },
      "narrow": {
        "1": "١",
        "2": "٢",
        "3": "٣",
        "4": "٤"
      },
      "wide": {
        "1": "الربع الأول",
        "2": "الربع الثاني",
        "3": "الربع الثالث",
        "4": "الربع الرابع"
      }
    },
    "stand-alone": {
      "abbreviated": {
        "1": "الربع الأول",
        "2": "الربع الثاني",
        "3": "الربع الثالث",
        "4": "الربع الرابع"
      },
      "narrow": {
        "1": "١",
        "2": "٢",
        "3": "٣",
        "4": "٤"
      },
      "wide": {
        "1": "الربع الأول",

```



```

        "2": "الربع الثاني",
        "3": "الربع الثالث",
        "4": "الربع الرابع"
    }
}
},
"dayPeriods": {
    "format": {
        "abbreviated": {
            "am": "ص",
            "pm": "م",
            "morning1": "فجراً",
            "morning2": "ص",
            "afternoon1": "ظهراً",
            "afternoon2": "بعد الظهر",
            "evening1": "مساءً",
            "night1": "منتصف الليل",
            "night2": "ل"
        },
        "narrow": {
            "am": "ص",
            "pm": "م",
            "morning1": "فجرًا",
            "morning2": "صباحًا",
            "afternoon1": "ظهراً",
            "afternoon2": "بعد الظهر",
            "evening1": "مساءً",
            "night1": "منتصف الليل",
            "night2": "ليلاً"
        },
        "wide": {
            "am": "ص",
            "pm": "م",
            "morning1": "فجرًا",
            "morning2": "صباحًا",
            "afternoon1": "ظهراً",
            "afternoon2": "بعد الظهر",
            "evening1": "مساءً",
            "night1": "منتصف الليل",
            "night2": "ليلاً"
        }
    },
    "stand-alone": {
        "abbreviated": {
            "am": "ص",
            "pm": "م",
            "morning1": "فجراً",
            "morning2": "ص",
            "afternoon1": "ظهراً",
            "afternoon2": "بعد الظهر",
            "evening1": "مساءً",
            "night1": "منتصف الليل",
            "night2": "ليلاً"
        },
        "narrow": {
            "am": "ص",
            "pm": "م",

```

```

        "morning1": "فجرًا",
        "morning2": "صباحًا",
        "afternoon1": "ظهرًا",
        "afternoon2": "بعد الظهر",
        "evening1": "مساءً",
        "night1": "منتصف الليل",
        "night2": "ليلاً"
    },
    "wide": {
        "am": "صباحًا",
        "pm": "مساءً",
        "morning1": "فجرًا",
        "morning2": "صباحًا",
        "afternoon1": "ظهرًا",
        "afternoon2": "بعد الظهر",
        "evening1": "مساءً",
        "night1": "منتصف الليل",
        "night2": "ليلاً"
    }
},
"eras": {
    "eraNames": {
        "0": "قبل الميلاد",
        "0-alt-variant": "BCE",
        "1": "ميلادي",
        "1-alt-variant": "بعد الميلاد"
    },
    "eraAbbr": {
        "0": "ق.م",
        "0-alt-variant": "BCE",
        "1": "م",
        "1-alt-variant": "ب.م"
    },
    "eraNarrow": {
        "0": "ق.م",
        "0-alt-variant": "BCE",
        "1": "م",
        "1-alt-variant": "ب.م"
    }
},
"dateFormats": {
    "full": "EEEE، d MMMM، y",
    "long": "d MMMM، y",
    "medium": "dd/MM/y",
    "short": "d/M/y"
},
"timeFormats": {
    "full": "h:mm:ss a zzzz",
    "long": "h:mm:ss a z",
    "medium": "h:mm:ss a",
    "short": "h:mm a"
},
"dateTimeFormats": {
    "full": "{1} {0}",
    "long": "{1} {0}",
    "medium": "{1} {0}",

```

```

"short": "{1} {0}",
"availableFormats": {
  "d": "d",
  "E": "ccc",
  "Ed": "E, d",
  "Ehm": "E h:mm a",
  "EHm": "E HH:mm",
  "Ehms": "E h:mm:ss a",
  "EHms": "E HH:mm:ss",
  "Gy": "y G",
  "GyMMM": "MMM y G",
  "GyMMMd": "d MMM, y G",
  "GyMMMEd": "E, d MMM, y G",
  "h": "h a",
  "H": "HH",
  "hm": "h:mm a",
  "Hm": "HH:mm",
  "hms": "h:mm:ss a",
  "Hms": "HH:mm:ss",
  "hmsv": "h:mm:ss a v",
  "Hmsv": "HH:mm:ss v",
  "hmv": "h:mm a v",
  "Hmv": "HH:mm v",
  "M": "L",
  "Md": "d/M",
  "MED": "E, d/M",
  "MMdd": "dd/MM",
  "MMM": "LLL",
  "MMMd": "d MMM",
  "MMMEd": "E, d MMM",
  "MMMMd": "d MMMM",
  "MMMMEd": "E, d MMMM",
  "MMMMW": "الأسبوع W من MMM",
  "MMMMW": "الأسبوع W من MMM",
  "MMMMW": "الأسبوع W من MMM",
  "MMMMW": "الأسبوع W من MMM",
  "MMMMW": "الأسبوع W من MMM",
  "MMMMW": "الأسبوع W من MMM",
  "ms": "mm:ss",
  "y": "y",
  "yM": "M/y",
  "yMd": "d/M/y",
  "yMED": "E, d/M/y",
  "yMM": "MM/y",
  "yMMM": "MMM y",
  "yMMMd": "d MMM, y",
  "yMMMEd": "E, d MMM, y",
  "yMMMM": "MMMM y",
  "yQQQ": "QQQ y",
  "yQQQQ": "QQQQ y",
  "yw": "الأسبوع w من سنة y",
  "yw": "الأسبوع w من سنة y",
  "yw": "الأسبوع w من سنة y",
  "yw": "الأسبوع w من سنة y",
  "yw": "الأسبوع w من سنة y",
  "yw": "الأسبوع w من سنة y"
},

```

```

"appendItems": {
  "Day": "{0} ({2}: {1})",
  "Day-Of-Week": "{0} {1}",
  "Era": "{1} {0}",
  "Hour": "{0} ({2}: {1})",
  "Minute": "{0} ({2}: {1})",
  "Month": "{0} ({2}: {1})",
  "Quarter": "{0} ({2}: {1})",
  "Second": "{0} ({2}: {1})",
  "Timezone": "{0} {1}",
  "Week": "{0} ({2}: {1})",
  "Year": "{1} {0}"
},
"intervalFormats": {
  "intervalFormatFallback": "{0} - {1}",
  "d": {
    "d": "d-d"
  },
  "h": {
    "a": "h a - h a",
    "h": "h-h a"
  },
  "H": {
    "H": "HH-HH"
  },
  "hm": {
    "a": "h:mm a - h:mm a",
    "h": "h:mm-h:mm a",
    "m": "h:mm-h:mm a"
  },
  "Hm": {
    "H": "HH:mm-HH:mm",
    "m": "HH:mm-HH:mm"
  },
  "hmv": {
    "a": "h:mm a - h:mm a v",
    "h": "h:mm-h:mm a v",
    "m": "h:mm-h:mm a v"
  },
  "Hmv": {
    "H": "HH:mm-HH:mm v",
    "m": "HH:mm-HH:mm v"
  },
  "hv": {
    "a": "h a - h a v",
    "h": "h-h a v"
  },
  "Hv": {
    "H": "HH-HH v"
  },
  "M": {
    "M": "M-M"
  },
  "Md": {
    "d": "M/d - M/d",
    "M": "M/d - M/d"
  }
},

```

```

    "MEd": {
      "d": "E, d/M - E, d/M",
      "M": "E, d/M - E, d/M"
    },
    "MMM": {
      "M": "MMM-MMM"
    },
    "MMMd": {
      "d": "d-d MMM",
      "M": "d MMM - d MMM"
    },
    "MMMEd": {
      "d": "E, d - E, d MMM",
      "M": "E, d MMM - E, d MMM"
    },
    "MMMM": {
      "M": "LLLL-LLLL"
    },
    "Y": {
      "Y": "Y-Y"
    },
    "YM": {
      "M": "M/Y - M/Y",
      "Y": "M/Y - M/Y"
    },
    "YMd": {
      "d": "d/M/Y - d/M/Y",
      "M": "d/M/Y - d/M/Y",
      "Y": "d/M/Y - d/M/Y"
    },
    "YMEd": {
      "d": "E, dd/MM/Y - E, dd/MM/Y",
      "M": "E, d/M/Y - E, d/M/Y",
      "Y": "E, d/M/Y - E, d/M/Y"
    },
    "YMMM": {
      "M": "MMM - MMM, Y",
      "Y": "MMM, Y - MMM, Y"
    },
    "YMMMd": {
      "d": "d-d MMM, Y",
      "M": "d MMM - d MMM, Y",
      "Y": "d MMM, Y - d MMM, Y"
    },
    "YMMMEd": {
      "d": "E, d - E, d MMM, Y",
      "M": "E, d MMM - E, d MMM, Y",
      "Y": "E, d MMM, Y - E, d MMM, Y"
    },
    "YMMMM": {
      "M": "MMMM - MMMM, Y",
      "Y": "MMMM, Y - MMMM, Y"
    }
  }
}
}
}
}

```

```

    }
  }
}

```

CA-GREGORIAN.JSX

```

{
  "main";
  {
    "ar";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 12879 $",
          "_cldrVersion";
          "30.0.3";
        }
        "language";
        "ar";
      }
      "dates";
      {
        "calendars";
        {
          "gregorian";
          {
            "months";
            {
              "format";
              {
                "abbreviated";
                {
                  "1";
                  "يناير",
                  "2";
                  "فبراير",
                  "3";
                  "مارس",
                  "4";
                  "أبريل",
                  "5";
                  "مايو",
                  "6";
                  "يونيو",
                  "7";
                  "يوليو",
                  "8";
                  "أغسطس",
                  "9";
                  "سبتمبر",
                  "10";
                  "أكتوبر",

```

```

        "11";
        "نوفمبر",
        "12";
        "ديسمبر";
    }
    "narrow";
    {
        "1";
        "ي",
        "2";
        "ف",
        "3";
        "م",
        "4";
        "أ",
        "5";
        "و",
        "6";
        "ن",
        "7";
        "ل",
        "8";
        "غ",
        "9";
        "س",
        "10";
        "ك",
        "11";
        "ب",
        "12";
        "د";
    }
    "wide";
    {
        "1";
        "يناير",
        "2";
        "فبراير",
        "3";
        "مارس",
        "4";
        "أبريل",
        "5";
        "مايو",
        "6";
        "يونيو",
        "7";
        "يوليو",
        "8";
        "أغسطس",
        "9";
        "سبتمبر",
        "10";
        "أكتوبر",
        "11";
        "نوفمبر",
        "12";
    }

```

```

        "ديسمبر";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
        "يناير",
        "2";
        "فبراير",
        "3";
        "مارس",
        "4";
        "أبريل",
        "5";
        "مايو",
        "6";
        "يونيو",
        "7";
        "يوليو",
        "8";
        "أغسطس",
        "9";
        "سبتمبر",
        "10";
        "أكتوبر",
        "11";
        "نوفمبر",
        "12";
        "ديسمبر";
    }
}
"narrow";
{
    "1";
    "ي",
    "2";
    "ف",
    "3";
    "م",
    "4";
    "أ",
    "5";
    "و",
    "6";
    "ن",
    "7";
    "ل",
    "8";
    "غ",
    "9";
    "س",
    "10";
    "ك",
    "11";
    "ب",
    "12";
}

```



```

        "د";
    }
    "wide";
    {
        "1";
        "يناير",
        "2";
        "فبراير",
        "3";
        "مارس",
        "4";
        "أبريل",
        "5";
        "مايو",
        "6";
        "يونيو",
        "7";
        "يوليو",
        "8";
        "أغسطس",
        "9";
        "سبتمبر",
        "10";
        "أكتوبر",
        "11";
        "نوفمبر",
        "12";
        "ديسمبر";
    }
}
"days";
{
    "format";
    {
        "abbreviated";
        {
            "sun";
            "الأحد",
            "mon";
            "الاثنين",
            "tue";
            "الثلاثاء",
            "wed";
            "الأربعاء",
            "thu";
            "الخميس",
            "fri";
            "الجمعة",
            "sat";
            "السبت";
        }
        "narrow";
        {
            "sun";
            "ح",
            "mon";

```

```

        "ن",
        "tue";
        "ث",
        "wed";
        "ر",
        "thu";
        "خ",
        "fri";
        "ج",
        "sat";
        "س";
    }
    "short";
    {
        "sun";
        "الأحد",
        "mon";
        "الاثنين",
        "tue";
        "الثلاثاء",
        "wed";
        "الأربعاء",
        "thu";
        "الخميس",
        "fri";
        "الجمعة",
        "sat";
        "السبت";
    }
    "wide";
    {
        "sun";
        "الأحد",
        "mon";
        "الاثنين",
        "tue";
        "الثلاثاء",
        "wed";
        "الأربعاء",
        "thu";
        "الخميس",
        "fri";
        "الجمعة",
        "sat";
        "السبت";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "sun";
        "الأحد",
        "mon";
        "الاثنين",
        "tue";
        "الثلاثاء",

```

```

        "wed";
        "الأربعاء",
        "thu";
        "الخميس",
        "fri";
        "الجمعة",
        "sat";
        "السبت";
    }
    "narrow";
    {
        "sun";
        "ح",
        "mon";
        "ن",
        "tue";
        "ث",
        "wed";
        "ر",
        "thu";
        "خ",
        "fri";
        "ج",
        "sat";
        "س";
    }
    "short";
    {
        "sun";
        "الأحد",
        "mon";
        "الاثنين",
        "tue";
        "الثلاثاء",
        "wed";
        "الأربعاء",
        "thu";
        "الخميس",
        "fri";
        "الجمعة",
        "sat";
        "السبت";
    }
    "wide";
    {
        "sun";
        "الأحد",
        "mon";
        "الاثنين",
        "tue";
        "الثلاثاء",
        "wed";
        "الأربعاء",
        "thu";
        "الخميس",
        "fri";
        "الجمعة",

```

```

        "sat";
        "السبت";
    }
}
}
"quarters";
{
    "format";
    {
        "abbreviated";
        {
            "1";
            "الربع الأول",
            "2";
            "الربع الثاني",
            "3";
            "الربع الثالث",
            "4";
            "الربع الرابع";
        }
        "narrow";
        {
            "1";
            "١",
            "2";
            "٢",
            "3";
            "٣",
            "4";
            "٤";
        }
        "wide";
        {
            "1";
            "الربع الأول",
            "2";
            "الربع الثاني",
            "3";
            "الربع الثالث",
            "4";
            "الربع الرابع";
        }
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
        "الربع الأول",
        "2";
        "الربع الثاني",
        "3";
        "الربع الثالث",
        "4";
        "الربع الرابع";
    }
    "narrow";

```

```

        {
            "1";
            "١",
            "2";
            "٢",
            "3";
            "٣",
            "4";
            "٤";
        }
        "wide";
        {
            "1";
            "الربع الأول",
            "2";
            "الربع الثاني",
            "3";
            "الربع الثالث",
            "4";
            "الربع الرابع";
        }
    }
}
"dayPeriods";
{
    "format";
    {
        "abbreviated";
        {
            "am";
            "ص",
            "pm";
            "م",
            "morning1";
            "فجراً",
            "morning2";
            "ص",
            "afternoon1";
            "ظهراً",
            "afternoon2";
            "بعد الظهر",
            "evening1";
            "مساءً",
            "night1";
            "منتصف الليل",
            "night2";
            "ل";
        }
        "narrow";
        {
            "am";
            "ص",
            "pm";
            "م",
            "morning1";
            "فجراً",
            "morning2";
        }
    }
}

```

```

        "صباحًا",
        "afternoon1";
        "ظهرًا",
        "afternoon2";
        "بعد الظهر",
        "evening1";
        "مساءً",
        "night1";
        "منتصف الليل",
        "night2";
        "ليلاً";
    }
    "wide";
    {
        "am";
        "ص",
        "pm";
        "م",
        "morning1";
        "فجرًا",
        "morning2";
        "صباحًا",
        "afternoon1";
        "ظهرًا",
        "afternoon2";
        "بعد الظهر",
        "evening1";
        "مساءً",
        "night1";
        "منتصف الليل",
        "night2";
        "ليلاً";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "am";
        "ص",
        "pm";
        "م",
        "morning1";
        "فجرًا",
        "morning2";
        "ص",
        "afternoon1";
        "ظهرًا",
        "afternoon2";
        "بعد الظهر",
        "evening1";
        "مساءً",
        "night1";
        "منتصف الليل",
        "night2";
        "ليلاً";
    }
}

```

```

        "narrow";
        {
            "am";
            "ص",
            "pm";
            "م",
            "morning1";
            "فجرا",
            "morning2";
            "صباحًا",
            "afternoon1";
            "ظهراً",
            "afternoon2";
            "بعد الظهر",
            "evening1";
            "مساءً",
            "night1";
            "منتصف الليل",
            "night2";
            "ليلاً";
        }
        "wide";
        {
            "am";
            "صباحًا",
            "pm";
            "مساءً",
            "morning1";
            "فجراً",
            "morning2";
            "صباحاً",
            "afternoon1";
            "ظهراً",
            "afternoon2";
            "بعد الظهر",
            "evening1";
            "مساءً",
            "night1";
            "منتصف الليل",
            "night2";
            "ليلاً";
        }
    }
    "eras";
    {
        "eraNames";
        {
            "0";
            "قبل الميلاد",
            "0-alt-variant";
            "BCE",
            "1";
            "ميلادي",
            "1-alt-variant";
            "بعد الميلاد";
        }
    }
}

```

```

        "eraAbbr";
        {
            "0";
            "م.ق",
            "0-alt-variant";
            "BCE",
            "1";
            "م",
            "1-alt-variant";
            "م.ب";
        }
        "eraNarrow";
        {
            "0";
            "م.ق",
            "0-alt-variant";
            "BCE",
            "1";
            "م",
            "1-alt-variant";
            "م.ب";
        }
    }
    "dateFormats";
    {
        "full";
        "EEEE, d MMMM, y",
        "long";
        "d MMMM, y",
        "medium";
        "dd/MM/y",
        "short";
        "d/M/y";
    }
    "timeFormats";
    {
        "full";
        "h:mm:ss a zzzz",
        "long";
        "h:mm:ss a z",
        "medium";
        "h:mm:ss a",
        "short";
        "h:mm a";
    }
    "dateTimeFormats";
    {
        "full";
        "{1} {0}",
        "long";
        "{1} {0}",
        "medium";
        "{1} {0}",
        "short";
        "{1} {0}",
        "availableFormats";
        {

```



```

"d";
"d",
  "E";
"ccc",
  "Ed";
"E, d",
  "Ehm";
"E h:mm a",
  "EHm";
"E HH:mm",
  "Ehms";
"E h:mm:ss a",
  "EHms";
"E HH:mm:ss",
  "Gy";
"y G",
  "GyMMM";
"MMM y G",
  "GyMMMd";
"d MMM, y G",
  "GyMMMED";
"E, d MMM, y G",
  "h";
"h a",
  "H";
"HH",
  "hm";
"h:mm a",
  "Hm";
"HH:mm",
  "hms";
"h:mm:ss a",
  "Hms";
"HH:mm:ss",
  "hmsv";
"h:mm:ss a v",
  "Hmsv";
"HH:mm:ss v",
  "hmv";
"h:mm a v",
  "Hmv";
"HH:mm v",
  "M";
"L",
  "Md";
"d/M",
  "MED";
"E, d/M",
  "MMdd";
"dd/MM",
  "MMM";
"LLL",
  "MMMd";
"d MMM",
  "MMMED";
"E, d MMM",
  "MMMMd";

```

```

        "d MMMM",
        "MMMMEd";
        "E، d MMMM",
        "MMMMW";
        "الأسبوع من W الـ MMM",
        "MMMMW";
        "الأسبوع من W الـ MMM",
        "MMMMW";
        "الأسبوع من W الـ MMM",
        "MMMMW";
        "الأسبوع من W الـ MMM",
        "MMMMW";
        "الأسبوع من W الـ MMM",
        "MMMMW";
        "الأسبوع من W الـ MMM",
        "ms";
        "mm:ss",
        "y";
        "Y",
        "yM";
        "M/Y",
        "yMd";
        "d/M/Y",
        "yMEd";
        "E، d/M/Y",
        "yMM";
        "MM/Y",
        "yMMM";
        "MMM y",
        "yMMMd";
        "d MMM، y",
        "yMMMEd";
        "E، d MMM، y",
        "yMMMM";
        "MMMM y",
        "yQQQ";
        "QQQ y",
        "yQQQQ";
        "QQQQ y",
        "yw";
        "y من سنة w الـ الأسبوع",
        "yw";
        "y من سنة w الـ الأسبوع",
        "yw";
        "y من سنة w الـ الأسبوع",
        "yw";
        "y من سنة w الـ الأسبوع",
        "yw";
        "y من سنة w الـ الأسبوع",
        "yw";
        "y من سنة w الـ الأسبوع";
    }
    "appendItems";
    {
        "Day";
        "{0} ({2}: {1})",
        "Day-Of-Week";
    }

```

```

    "{0} {1}",
    "Era";
    "{1} {0}",
    "Hour";
    "{0} ({2}: {1})",
    "Minute";
    "{0} ({2}: {1})",
    "Month";
    "{0} ({2}: {1})",
    "Quarter";
    "{0} ({2}: {1})",
    "Second";
    "{0} ({2}: {1})",
    "Timezone";
    "{0} {1}",
    "Week";
    "{0} ({2}: {1})",
    "Year";
    "{1} {0}";
  }
  "intervalFormats";
  {
    "intervalFormatFallback";
    "{0} - {1}",
    "d";
    {
      "d";
      "d-d";
    }
    "h";
    {
      "a";
      "h a - h a",
      "h";
      "h-h a";
    }
    "H";
    {
      "H";
      "HH-HH";
    }
    "hm";
    {
      "a";
      "h:mm a - h:mm a",
      "h";
      "h:mm-h:mm a",
      "m";
      "h:mm-h:mm a";
    }
    "Hm";
    {
      "H";
      "HH:mm-HH:mm",
      "m";
      "HH:mm-HH:mm";
    }
  }

```

```

"hmV";
{
    "a";
    "h:mm a - h:mm a v",
    "h";
    "h:mm-h:mm a v",
    "m";
    "h:mm-h:mm a v";
}
"HmV";
{
    "H";
    "HH:mm-HH:mm v",
    "m";
    "HH:mm-HH:mm v";
}
"hv";
{
    "a";
    "h a - h a v",
    "h";
    "h-h a v";
}
"Hv";
{
    "H";
    "HH-HH v";
}
"M";
{
    "M";
    "M-M";
}
"Md";
{
    "d";
    "M/d - M/d",
    "M";
    "M/d - M/d";
}
"MEd";
{
    "d";
    "E, d/M - E, d/M",
    "M";
    "E, d/M - E, d/M";
}
"MMM";
{
    "M";
    "MMM-MMM";
}
"MMMd";
{
    "d";
    "d-d MMM",
    "M";

```

```

        "d MMM - d MMM";
    }
    "MMMEd";
    {
        "d";
        "E, d - E, d MMM",
        "M";
        "E, d MMM - E, d MMM";
    }
    "MMMM";
    {
        "M";
        "LLLL-LLLL";
    }
    "Y";
    {
        "Y";
        "Y-Y";
    }
    "yM";
    {
        "M";
        "M/Y - M/Y",
        "Y";
        "M/Y - M/Y";
    }
    "yMd";
    {
        "d";
        "d/M/Y - d/M/Y",
        "M";
        "d/M/Y - d/M/Y",
        "Y";
        "d/M/Y - d/M/Y";
    }
    "yMEd";
    {
        "d";
        "E, dd/MM/Y - E, dd/MM/Y",
        "M";
        "E, d/M/Y - E, d/M/Y",
        "Y";
        "E, d/M/Y - E, d/M/Y";
    }
    "yMMM";
    {
        "M";
        "MMM - MMM, Y",
        "Y";
        "MMM, Y - MMM, Y";
    }
    "yMMMd";
    {
        "d";
        "d-d MMM, Y",
        "M";
        "d MMM - d MMM, Y",

```

```

    "Y";
    "d MMM. y - d MMM. y";
}
"yMMMEd";
{
    "d";
    "E. d - E. d MMM. y",
    "M";
    "E. d MMM - E. d MMM. y",
    "Y";
    "E. d MMM. y - E. d MMM. y";
}
"yMMMM";
{
    "M";
    "MMMM - MMMM. y",
    "Y";
    "MMMM. y - MMMM. y";
}
}
}
}
}
}
}
}
}
}

```

CURRENCIES.JSON

```
{
  "main": {
    "ar": {
      "identity": {
        "version": {
          "_number": "$Revision: 12879 $",
          "_cldrVersion": "30.0.3"
        },
        "language": "ar"
      },
      "numbers": {
        "currencies": {
          "ADP": {
            "displayName": "بيستا أندوري",
            "symbol": "ADP"
          },
          "AED": {
            "displayName": "درهم إماراتي",
            "displayName-count-zero": "درهم إماراتي",
            "displayName-count-one": "درهم إماراتي",
            "displayName-count-two": "درهم إماراتي",
            "displayName-count-few": "درهم إماراتي",
            "displayName-count-many": "درهم إماراتي",
            "displayName-count-other": "درهم إماراتي",
            "symbol": "د.إ."
          }
        }
      }
    }
  }
}
```

```

"AFA": {
  "displayName": "2002-1927 - أفغاني",
  "symbol": "AFA"
},
"AFN": {
  "displayName": "أفغاني",
  "displayName-count-zero": "أفغاني أفغانستان",
  "displayName-count-one": "أفغاني أفغانستان",
  "displayName-count-two": "أفغاني أفغانستان",
  "displayName-count-few": "أفغاني أفغانستان",
  "displayName-count-many": "أفغاني أفغانستان",
  "displayName-count-other": "أفغاني أفغانستان",
  "symbol": "AFN"
},
"ALL": {
  "displayName": "ليك ألباني",
  "displayName-count-zero": "ليك ألباني",
  "displayName-count-one": "ليك ألباني",
  "displayName-count-two": "ليك ألباني",
  "displayName-count-few": "ليك ألباني",
  "displayName-count-many": "ليك ألباني",
  "displayName-count-other": "ليك ألباني",
  "symbol": "ALL"
},
"AMD": {
  "displayName": "درام أرميني",
  "displayName-count-zero": "درام أرميني",
  "displayName-count-one": "درام أرميني",
  "displayName-count-two": "درام أرميني",
  "displayName-count-few": "درام أرميني",
  "displayName-count-many": "درام أرميني",
  "displayName-count-other": "درام أرميني",
  "symbol": "AMD"
},
"ANG": {
  "displayName": "غيلدر أنتيلي هولندي",
  "displayName-count-zero": "غيلدر أنتيلي هولندي",
  "displayName-count-one": "غيلدر أنتيلي هولندي",
  "displayName-count-two": "غيلدر أنتيلي هولندي",
  "displayName-count-few": "غيلدر أنتيلي هولندي",
  "displayName-count-many": "غيلدر أنتيلي هولندي",
  "displayName-count-other": "غيلدر أنتيلي هولندي",
  "symbol": "ANG"
},
"AOA": {
  "displayName": "كوانزا أنجولي",
  "displayName-count-zero": "كوانزا أنجولي",
  "displayName-count-one": "كوانزا أنجولي",
  "displayName-count-two": "كوانزا أنجولي",
  "displayName-count-few": "كوانزا أنجولي",
  "displayName-count-many": "كوانزا أنجولي",
  "displayName-count-other": "كوانزا أنجولي",
  "symbol": "AOA",
  "symbol-alt-narrow": "Kz"
},
"AOK": {
  "displayName": "كوانزا أنجولي - 1990-1977",

```

```

    "symbol": "AOK"
  },
  "AON": {
    "displayName": "كوانزا أنجولي جديدة - 2000-1990",
    "symbol": "AON"
  },
  "AOR": {
    "displayName": "كوانزا أنجولي معدلة - 1999 - 1995",
    "symbol": "AOR"
  },
  "ARA": {
    "displayName": "استرال أرجنتيني",
    "symbol": "ARA"
  },
  "ARL": {
    "displayName": "ARL",
    "symbol": "ARL"
  },
  "ARM": {
    "displayName": "ARM",
    "symbol": "ARM"
  },
  "ARP": {
    "displayName": "بيزو أرجنتيني - 1985-1983",
    "symbol": "ARP"
  },
  "ARS": {
    "displayName": "بيزو أرجنتيني",
    "displayName-count-zero": "بيزو أرجنتيني",
    "displayName-count-one": "بيزو أرجنتيني",
    "displayName-count-two": "بيزو أرجنتيني",
    "displayName-count-few": "بيزو أرجنتيني",
    "displayName-count-many": "بيزو أرجنتيني",
    "displayName-count-other": "بيزو أرجنتيني",
    "symbol": "ARS",
    "symbol-alt-narrow": "AR$"
  },
  "ATS": {
    "displayName": "شلن نمساوي",
    "symbol": "ATS"
  },
  "AUD": {
    "displayName": "دولار أسترالي",
    "displayName-count-zero": "دولار أسترالي",
    "displayName-count-one": "دولار أسترالي",
    "displayName-count-two": "دولار أسترالي",
    "displayName-count-few": "دولار أسترالي",
    "displayName-count-many": "دولار أسترالي",
    "displayName-count-other": "دولار أسترالي",
    "symbol": "AU$",
    "symbol-alt-narrow": "AU$"
  },
  "AWG": {
    "displayName": "فلورن أروبي",
    "displayName-count-zero": "فلورن أروبي",
    "displayName-count-one": "فلورن أروبي",
    "displayName-count-two": "فلورن أروبي",

```



```

    "displayName-count-few": "فلورن أروبي",
    "displayName-count-many": "فلورن أروبي",
    "displayName-count-other": "فلورن أروبي",
    "symbol": "AWG"
  },
  "AZM": {
    "displayName": "مانات أذربيجاني",
    "symbol": "AZM"
  },
  "AZN": {
    "displayName": "مانات أذربيجان",
    "displayName-count-zero": "مانت أذربيجاني",
    "displayName-count-one": "مانت أذربيجاني",
    "displayName-count-two": "مانت أذربيجاني",
    "displayName-count-few": "مانت أذربيجاني",
    "displayName-count-many": "مانت أذربيجاني",
    "displayName-count-other": "مانت أذربيجاني",
    "symbol": "AZN"
  },
  "BAD": {
    "displayName": "دينار البوسنة والهرسك",
    "symbol": "BAD"
  },
  "BAM": {
    "displayName": "مارك البوسنة والهرسك قابل للتحويل",
    "displayName-count-zero": "مارك البوسنة والهرسك قابل للتحويل",
    "displayName-count-one": "مارك البوسنة والهرسك قابل للتحويل",
    "displayName-count-two": "مارك البوسنة والهرسك قابل للتحويل",
    "displayName-count-few": "مارك البوسنة والهرسك قابل للتحويل",
    "displayName-count-many": "مارك البوسنة والهرسك قابل للتحويل",
    "displayName-count-other": "مارك البوسنة والهرسك قابل للتحويل",
    "symbol": "BAM",
    "symbol-alt-narrow": "KM"
  },
  "BAN": {
    "displayName": "BAN",
    "symbol": "BAN"
  },
  "BBD": {
    "displayName": "دولار بربادوسي",
    "displayName-count-zero": "دولار بربادوسي",
    "displayName-count-one": "دولار بربادوسي",
    "displayName-count-two": "دولار بربادوسي",
    "displayName-count-few": "دولار بربادوسي",
    "displayName-count-many": "دولار بربادوسي",
    "displayName-count-other": "دولار بربادوسي",
    "symbol": "BBD",
    "symbol-alt-narrow": "BB$"
  },
  "BDT": {
    "displayName": "تাকা بنجلاديشي",
    "displayName-count-zero": "تাকা بنجلاديشي",
    "displayName-count-one": "تাকা بنجلاديشي",
    "displayName-count-two": "تাকা بنجلاديشي",
    "displayName-count-few": "تাকা بنجلاديشي",
    "displayName-count-many": "تাকা بنجلاديشي",
    "displayName-count-other": "تাকা بنجلاديشي",

```

```

        "symbol": "BDT",
        "symbol-alt-narrow": "৳"
    },
    "BEC": {
        "displayName": "فرنك بلجيكي قابل للتحويل",
        "symbol": "BEC"
    },
    "BEF": {
        "displayName": "فرنك بلجيكي",
        "symbol": "BEF"
    },
    "BEL": {
        "displayName": "فرنك بلجيكي مالي",
        "symbol": "BEL"
    },
    "BGL": {
        "displayName": "BGL",
        "symbol": "BGL"
    },
    "BGM": {
        "displayName": "BGM",
        "symbol": "BGM"
    },
    "BGN": {
        "displayName": "ليف بلغاري",
        "displayName-count-zero": "ليف بلغاري",
        "displayName-count-one": "ليف بلغاري",
        "displayName-count-two": "ليف بلغاري",
        "displayName-count-few": "ليف بلغاري",
        "displayName-count-many": "ليف بلغاري",
        "displayName-count-other": "ليف بلغاري",
        "symbol": "BGN"
    },
    "BGO": {
        "displayName": "BGO",
        "symbol": "BGO"
    },
    "BHD": {
        "displayName": "دينار بحريني",
        "displayName-count-zero": "دينار بحريني",
        "displayName-count-one": "دينار بحريني",
        "displayName-count-two": "دينار بحريني",
        "displayName-count-few": "دينار بحريني",
        "displayName-count-many": "دينار بحريني",
        "displayName-count-other": "دينار بحريني",
        "symbol": "د.ب.د"
    },
    "BIF": {
        "displayName": "فرنك بروندي",
        "displayName-count-zero": "فرنك بروندي",
        "displayName-count-one": "فرنك بروندي",
        "displayName-count-two": "فرنك بروندي",
        "displayName-count-few": "فرنك بروندي",
        "displayName-count-many": "فرنك بروندي",
        "displayName-count-other": "فرنك بروندي",
        "symbol": "BIF"
    }

```

```

    },
    "BMD": {
      "displayName": "دولار برمودي",
      "displayName-count-zero": "دولار برمودي",
      "displayName-count-one": "دولار برمودي",
      "displayName-count-two": "دولار برمودي",
      "displayName-count-few": "دولار برمودي",
      "displayName-count-many": "دولار برمودي",
      "displayName-count-other": "دولار برمودي",
      "symbol": "BMD",
      "symbol-alt-narrow": "BM$"
    },
    "BND": {
      "displayName": "دولار برونائي",
      "displayName-count-zero": "دولار برونائي",
      "displayName-count-one": "دولار برونائي",
      "displayName-count-two": "دولار برونائي",
      "displayName-count-few": "دولار برونائي",
      "displayName-count-many": "دولار برونائي",
      "displayName-count-other": "دولار برونائي",
      "symbol": "BND",
      "symbol-alt-narrow": "BN$"
    },
    "BOB": {
      "displayName": "بوليفيانو بوليفي",
      "displayName-count-zero": "بوليفيانو بوليفي",
      "displayName-count-one": "بوليفيانو بوليفي",
      "displayName-count-two": "بوليفيانو بوليفي",
      "displayName-count-few": "بوليفيانو بوليفي",
      "displayName-count-many": "بوليفيانو بوليفي",
      "displayName-count-other": "بوليفيانو بوليفي",
      "symbol": "BOB",
      "symbol-alt-narrow": "Bs"
    },
    "BOL": {
      "displayName": "BOL",
      "symbol": "BOL"
    },
    "BOP": {
      "displayName": "بيزو بوليفي",
      "symbol": "BOP"
    },
    "BOV": {
      "displayName": "مفدول بوليفي",
      "symbol": "BOV"
    },
    "BRB": {
      "displayName": "نوفو كروزايرو برازيل - 1986-1967",
      "symbol": "BRB"
    },
    "BRC": {
      "displayName": "كروزادو برازيل",
      "symbol": "BRC"
    },
    "BRE": {
      "displayName": "كروزايرو برازيل - 1993-1990",
      "symbol": "BRE"
    }
  }

```

```

    },
    "BRL": {
      "displayName": "ريال برازيلی",
      "displayName-count-zero": "ريال برازيلی",
      "displayName-count-one": "ريال برازيلی",
      "displayName-count-two": "ريال برازيلی",
      "displayName-count-few": "ريال برازيلی",
      "displayName-count-many": "ريال برازيلی",
      "displayName-count-other": "ريال برازيلی",
      "symbol": "R$",
      "symbol-alt-narrow": "R$"
    },
    "BRN": {
      "displayName": "BRN",
      "symbol": "BRN"
    },
    "BRR": {
      "displayName": "BRR",
      "symbol": "BRR"
    },
    "BRZ": {
      "displayName": "BRZ",
      "symbol": "BRZ"
    },
    "BSD": {
      "displayName": "دولار باهامی",
      "displayName-count-zero": "دولار باهامی",
      "displayName-count-one": "دولار باهامی",
      "displayName-count-two": "دولار باهامی",
      "displayName-count-few": "دولار باهامی",
      "displayName-count-many": "دولار باهامی",
      "displayName-count-other": "دولار باهامی",
      "symbol": "BSD",
      "symbol-alt-narrow": "BS$"
    },
    "BTN": {
      "displayName": "نولتوم بوتانی",
      "displayName-count-zero": "نولتوم بوتانی",
      "displayName-count-one": "نولتوم بوتانی",
      "displayName-count-two": "نولتوم بوتانی",
      "displayName-count-few": "نولتوم بوتانی",
      "displayName-count-many": "نولتوم بوتانی",
      "displayName-count-other": "نولتوم بوتانی",
      "symbol": "BTN"
    },
    "BUK": {
      "displayName": "کیات بورمی",
      "symbol": "BUK"
    },
    "BWP": {
      "displayName": "پولا بتسوانی",
      "displayName-count-zero": "پولا بتسوانی",
      "displayName-count-one": "پولا بتسوانی",
      "displayName-count-two": "پولا بتسوانی",
      "displayName-count-few": "پولا بتسوانی",
      "displayName-count-many": "پولا بتسوانی",
      "displayName-count-other": "پولا بتسوانی",

```

```

    "symbol": "BWP",
    "symbol-alt-narrow": "P"
  },
  "BYB": {
    "displayName": "روبل بيلاروسي جديد - 1999-1994",
    "symbol": "BYB"
  },
  "BYN": {
    "displayName": "روبل بيلاروسي",
    "displayName-count-zero": "روبل بيلاروسي",
    "displayName-count-one": "روبل بيلاروسي",
    "displayName-count-two": "روبل بيلاروسي",
    "displayName-count-few": "روبل بيلاروسي",
    "displayName-count-many": "روبل بيلاروسي",
    "displayName-count-other": "روبل بيلاروسي",
    "symbol": "BYN",
    "symbol-alt-narrow": "p."
  },
  "BYR": {
    "displayName": " ) ۲۰۱۶-۲۰۰۰ ( روبل بيلاروسي",
    "displayName-count-zero": " ) ۲۰۱۶-۲۰۰۰ ( روبل بيلاروسي",
    "displayName-count-one": " ) ۲۰۱۶-۲۰۰۰ ( روبل بيلاروسي",
    "displayName-count-two": " ) ۲۰۱۶-۲۰۰۰ ( روبل بيلاروسي",
    "displayName-count-few": " ) ۲۰۱۶-۲۰۰۰ ( روبل بيلاروسي",
    "displayName-count-many": " ) ۲۰۱۶-۲۰۰۰ ( روبل بيلاروسي",
    "displayName-count-other": " ) ۲۰۱۶-۲۰۰۰ ( روبل بيلاروسي",
    "symbol": "BYR"
  },
  "BZD": {
    "displayName": "دولار بليزي",
    "displayName-count-zero": "دولار بليزي",
    "displayName-count-one": "دولار بليزي",
    "displayName-count-two": "دولاران بليزيان",
    "displayName-count-few": "دولار بليزي",
    "displayName-count-many": "دولار بليزي",
    "displayName-count-other": "دولار بليزي",
    "symbol": "BZD",
    "symbol-alt-narrow": "BZ$"
  },
  "CAD": {
    "displayName": "دولار كندي",
    "displayName-count-zero": "دولار كندي",
    "displayName-count-one": "دولار كندي",
    "displayName-count-two": "دولار كندي",
    "displayName-count-few": "دولار كندي",
    "displayName-count-many": "دولار كندي",
    "displayName-count-other": "دولار كندي",
    "symbol": "CA$",
    "symbol-alt-narrow": "CA$"
  },
  "CDF": {
    "displayName": "فرنك كونغولي",
    "displayName-count-zero": "فرنك كونغولي",
    "displayName-count-one": "فرنك كونغولي",
    "displayName-count-two": "فرنك كونغولي",
    "displayName-count-few": "فرنك كونغولي",
    "displayName-count-many": "فرنك كونغولي",

```

```

        "displayName-count-other": "فرنك كونغولي",
        "symbol": "CDF"
    },
    "CHE": {
        "displayName": "CHE",
        "symbol": "CHE"
    },
    "CHF": {
        "displayName": "فرنك سويسري",
        "displayName-count-zero": "فرنك سويسري",
        "displayName-count-one": "فرنك سويسري",
        "displayName-count-two": "فرنك سويسري",
        "displayName-count-few": "فرنك سويسري",
        "displayName-count-many": "فرنك سويسري",
        "displayName-count-other": "فرنك سويسري",
        "symbol": "CHF"
    },
    "CHW": {
        "displayName": "CHW",
        "symbol": "CHW"
    },
    "CLE": {
        "displayName": "CLE",
        "symbol": "CLE"
    },
    "CLF": {
        "displayName": "CLF",
        "symbol": "CLF"
    },
    "CLP": {
        "displayName": "بيزو شيلي",
        "displayName-count-zero": "بيزو شيلي",
        "displayName-count-one": "بيزو شيلي",
        "displayName-count-two": "بيزو شيلي",
        "displayName-count-few": "بيزو شيلي",
        "displayName-count-many": "بيزو شيلي",
        "displayName-count-other": "بيزو شيلي",
        "symbol": "CLP",
        "symbol-alt-narrow": "CL$"
    },
    "CNY": {
        "displayName": "يوان صيني",
        "displayName-count-zero": "يوان صيني",
        "displayName-count-one": "يوان صيني",
        "displayName-count-two": "يوان صيني",
        "displayName-count-few": "يوان صيني",
        "displayName-count-many": "يوان صيني",
        "displayName-count-other": "يوان صيني",
        "symbol": "CN¥",
        "symbol-alt-narrow": "CN¥"
    },
    "COP": {
        "displayName": "بيزو كولومبي",
        "displayName-count-zero": "بيزو كولومبي",
        "displayName-count-one": "بيزو كولومبي",
        "displayName-count-two": "بيزو كولومبي",
        "displayName-count-few": "بيزو كولومبي",

```

```

        "displayName-count-many": "بيزو كولومبي",
        "displayName-count-other": "بيزو كولومبي",
        "symbol": "COP",
        "symbol-alt-narrow": "CO$"
    },
    "COU": {
        "displayName": "COU",
        "symbol": "COU"
    },
    "CRC": {
        "displayName": "كولن كوستا ريكي",
        "displayName-count-zero": "كولن كوستا ريكي",
        "displayName-count-one": "كولن كوستا ريكي",
        "displayName-count-two": "كولن كوستا ريكي",
        "displayName-count-few": "كولن كوستا ريكي",
        "displayName-count-many": "كولن كوستا ريكي",
        "displayName-count-other": "كولن كوستا ريكي",
        "symbol": "CRC",
        "symbol-alt-narrow": "₡"
    },
    "CSD": {
        "displayName": "دينار صربي قديم",
        "symbol": "CSD"
    },
    "CSK": {
        "displayName": "كرونه تشيكوسلوفاكيا",
        "symbol": "CSK"
    },
    "CUC": {
        "displayName": "بيزو كوبي قابل للتحويل",
        "displayName-count-zero": "بيزو كوبي قابل للتحويل",
        "displayName-count-one": "بيزو كوبي قابل للتحويل",
        "displayName-count-two": "بيزو كوبي قابل للتحويل",
        "displayName-count-few": "بيزو كوبي قابل للتحويل",
        "displayName-count-many": "بيزو كوبي قابل للتحويل",
        "displayName-count-other": "بيزو كوبي قابل للتحويل",
        "symbol": "CUC",
        "symbol-alt-narrow": "$"
    },
    "CUP": {
        "displayName": "بيزو كوبي",
        "displayName-count-zero": "بيزو كوبي",
        "displayName-count-one": "بيزو كوبي",
        "displayName-count-two": "بيزو كوبي",
        "displayName-count-few": "بيزو كوبي",
        "displayName-count-many": "بيزو كوبي",
        "displayName-count-other": "بيزو كوبي",
        "symbol": "CUP",
        "symbol-alt-narrow": "CU$"
    },
    "CVE": {
        "displayName": "اسكودو الرأس الخضراء",
        "displayName-count-zero": "اسكودو الرأس الخضراء",
        "displayName-count-one": "اسكودو الرأس الخضراء",
        "displayName-count-two": "اسكودو الرأس الخضراء",
        "displayName-count-few": "اسكودو الرأس الخضراء",
        "displayName-count-many": "اسكودو الرأس الخضراء",

```

```

    "displayName-count-other": "اسكودو الرأس الخضراء",
    "symbol": "CVE"
  },
  "CYP": {
    "displayName": "جنيه قبرصي",
    "symbol": "CYP"
  },
  "CZK": {
    "displayName": "كرونة تشيكية",
    "displayName-count-zero": "كرونة تشيكية",
    "displayName-count-one": "كرونة تشيكية",
    "displayName-count-two": "كرونة تشيكية",
    "displayName-count-few": "كرونة تشيكية",
    "displayName-count-many": "كرونة تشيكية",
    "displayName-count-other": "كرونة تشيكية",
    "symbol": "CZK",
    "symbol-alt-narrow": "Kč"
  },
  "DDM": {
    "displayName": "أوستمارك ألماني شرقي",
    "symbol": "DDM"
  },
  "DEM": {
    "displayName": "مارك ألماني",
    "symbol": "DEM"
  },
  "DJF": {
    "displayName": "فرنك جيبوتي",
    "displayName-count-zero": "فرنك جيبوتي",
    "displayName-count-one": "فرنك جيبوتي",
    "displayName-count-two": "فرنك جيبوتي",
    "displayName-count-few": "فرنك جيبوتي",
    "displayName-count-many": "فرنك جيبوتي",
    "displayName-count-other": "فرنك جيبوتي",
    "symbol": "DJF"
  },
  "DKK": {
    "displayName": "كرونة دانماركي",
    "displayName-count-zero": "كرونة دانماركي",
    "displayName-count-one": "كرونة دانماركي",
    "displayName-count-two": "كرونة دانماركي",
    "displayName-count-few": "كرونة دانماركي",
    "displayName-count-many": "كرونة دانماركي",
    "displayName-count-other": "كرونة دانماركي",
    "symbol": "DKK",
    "symbol-alt-narrow": "kr"
  },
  "DOP": {
    "displayName": "بيزو الدومنيكان",
    "displayName-count-zero": "بيزو الدومنيكان",
    "displayName-count-one": "بيزو الدومنيكان",
    "displayName-count-two": "بيزو الدومنيكان",
    "displayName-count-few": "بيزو الدومنيكان",
    "displayName-count-many": "بيزو الدومنيكان",
    "displayName-count-other": "بيزو الدومنيكان",
    "symbol": "DOP",
    "symbol-alt-narrow": "DO$"
  }

```



```

    },
    "DZD": {
      "displayName": "دينار جزائري",
      "displayName-count-zero": "دينار جزائري",
      "displayName-count-one": "دينار جزائري",
      "displayName-count-two": "ديناران جزائريان",
      "displayName-count-few": "دينارات جزائرية",
      "displayName-count-many": "دينارًا جزائريًا",
      "displayName-count-other": "دينار جزائري",
      "symbol": "د.ج."
    },
    "ECS": {
      "displayName": "ECS",
      "symbol": "ECS"
    },
    "ECV": {
      "displayName": "ECV",
      "symbol": "ECV"
    },
    "EEK": {
      "displayName": "كرونه استونية",
      "symbol": "EEK"
    },
    "EGP": {
      "displayName": "جنيه مصري",
      "displayName-count-zero": "جنيه مصري",
      "displayName-count-one": "جنيه مصري",
      "displayName-count-two": "جنيهان مصريان",
      "displayName-count-few": "جنيهات مصرية",
      "displayName-count-many": "جنيهاً مصريًا",
      "displayName-count-other": "جنيه مصري",
      "symbol": "ج.م.",
      "symbol-alt-narrow": "E£"
    },
    "ERN": {
      "displayName": "ناكفا أريتري",
      "displayName-count-zero": "ناكفا أريتري",
      "displayName-count-one": "ناكفا أريتري",
      "displayName-count-two": "ناكفا أريتري",
      "displayName-count-few": "ناكفا أريتري",
      "displayName-count-many": "ناكفا أريتري",
      "displayName-count-other": "ناكفا أريتري",
      "symbol": "ERN"
    },
    "ESA": {
      "displayName": "ESA",
      "symbol": "ESA"
    },
    "ESB": {
      "displayName": "ESB",
      "symbol": "ESB"
    },
    "ESP": {
      "displayName": "بيزيتا إسباني",
      "symbol": "ESP",
      "symbol-alt-narrow": "₧"
    },
  },

```

```

"ETB": {
  "displayName": "بیر اٹیوبی",
  "displayName-count-zero": "بیر اٹیوبی",
  "displayName-count-one": "بیر اٹیوبی",
  "displayName-count-two": "بیر اٹیوبی",
  "displayName-count-few": "بیر اٹیوبی",
  "displayName-count-many": "بیر اٹیوبی",
  "displayName-count-other": "بیر اٹیوبی",
  "symbol": "ETB"
},
"EUR": {
  "displayName": "یورو",
  "displayName-count-zero": "یورو",
  "displayName-count-one": "یورو",
  "displayName-count-two": "یورو",
  "displayName-count-few": "یورو",
  "displayName-count-many": "یورو",
  "displayName-count-other": "یورو",
  "symbol": "€",
  "symbol-alt-narrow": "€"
},
"FIM": {
  "displayName": "مارکا فنلندی",
  "symbol": "FIM"
},
"FJD": {
  "displayName": "دولار فیجی",
  "displayName-count-zero": "دولار فیجی",
  "displayName-count-one": "دولار فیجی",
  "displayName-count-two": "دولار فیجی",
  "displayName-count-few": "دولار فیجی",
  "displayName-count-many": "دولار فیجی",
  "displayName-count-other": "دولار فیجی",
  "symbol": "FJD",
  "symbol-alt-narrow": "FJ$"
},
"FKP": {
  "displayName": "جنيه جزر فوکلاند",
  "displayName-count-zero": "جنيه جزر فوکلاند",
  "displayName-count-one": "جنيه جزر فوکلاند",
  "displayName-count-two": "جنيه جزر فوکلاند",
  "displayName-count-few": "جنيه جزر فوکلاند",
  "displayName-count-many": "جنيه جزر فوکلاند",
  "displayName-count-other": "جنيه جزر فوکلاند",
  "symbol": "FKP",
  "symbol-alt-narrow": "£"
},
"FRF": {
  "displayName": "فرنك فرنسي",
  "symbol": "FRF"
},
"GBP": {
  "displayName": "جنيه إسترليني",
  "displayName-count-zero": "جنيه إسترليني",
  "displayName-count-one": "جنيه إسترليني",
  "displayName-count-two": "جنيه إسترليني",
  "displayName-count-few": "جنيه إسترليني",

```

```

    "displayName-count-many": "جنيه إسترليني",
    "displayName-count-other": "جنيه إسترليني",
    "symbol": "£",
    "symbol-alt-narrow": "UK£"
  },
  "GEK": {
    "displayName": "GEK",
    "symbol": "GEK"
  },
  "GEL": {
    "displayName": "ლარი جورجى",
    "displayName-count-zero": "لاري جورجى",
    "displayName-count-one": "لاري جورجى",
    "displayName-count-two": "لاري جورجى",
    "displayName-count-few": "لاري جورجى",
    "displayName-count-many": "لاري جورجى",
    "displayName-count-other": "لاري جورجى",
    "symbol": "GEL",
    "symbol-alt-narrow": "ლ",
    "symbol-alt-variant": "ლ"
  },
  "GHC": {
    "displayName": "سيدي غاني",
    "symbol": "GHC"
  },
  "GHS": {
    "displayName": "سيدي غانا",
    "displayName-count-zero": "سيدي غانا",
    "displayName-count-one": "سيدي غانا",
    "displayName-count-two": "سيدي غانا",
    "displayName-count-few": "سيدي غانا",
    "displayName-count-many": "سيدي غانا",
    "displayName-count-other": "سيدي غانا",
    "symbol": "GHS"
  },
  "GIP": {
    "displayName": "جنيه جبل طارق",
    "displayName-count-zero": "جنيه جبل طارق",
    "displayName-count-one": "جنيه جبل طارق",
    "displayName-count-two": "جنيه جبل طارق",
    "displayName-count-few": "جنيه جبل طارق",
    "displayName-count-many": "جنيه جبل طارق",
    "displayName-count-other": "جنيه جبل طارق",
    "symbol": "GIP",
    "symbol-alt-narrow": "£"
  },
  "GMD": {
    "displayName": "دلاسي جامبي",
    "displayName-count-zero": "دلاسي جامبي",
    "displayName-count-one": "دلاسي جامبي",
    "displayName-count-two": "دلاسي جامبي",
    "displayName-count-few": "دلاسي جامبي",
    "displayName-count-many": "دلاسي جامبي",
    "displayName-count-other": "دلاسي جامبي",
    "symbol": "GMD"
  },
  "GNF": {

```

```

      "displayName": "فرنك غينيا",
      "displayName-count-zero": "فرنك غينيا",
      "displayName-count-one": "فرنك غينيا",
      "displayName-count-two": "فرنك غينيا",
      "displayName-count-few": "فرنك غينيا",
      "displayName-count-many": "فرنك غينيا",
      "displayName-count-other": "فرنك غينيا",
      "symbol": "GNF",
      "symbol-alt-narrow": "FG"
    },
    "GNS": {
      "displayName": "سيللي غينيا",
      "symbol": "GNS"
    },
    "GQE": {
      "displayName": "اكويل جونيينا غينيا الاستوائية",
      "symbol": "GQE"
    },
    "GRD": {
      "displayName": "دراخما يوناني",
      "symbol": "GRD"
    },
    "GTQ": {
      "displayName": "كوتزال جواتيمالا",
      "displayName-count-zero": "كوتزال جواتيمالا",
      "displayName-count-one": "كوتزال جواتيمالا",
      "displayName-count-two": "كوتزال جواتيمالا",
      "displayName-count-few": "كوتزال جواتيمالا",
      "displayName-count-many": "كوتزال جواتيمالا",
      "displayName-count-other": "كوتزال جواتيمالا",
      "symbol": "GTQ",
      "symbol-alt-narrow": "Q"
    },
    "GWE": {
      "displayName": "اسكود برتغالي غينيا",
      "symbol": "GWE"
    },
    "GWP": {
      "displayName": "بيزو غينيا بيساو",
      "symbol": "GWP"
    },
    "GYD": {
      "displayName": "دولار غيانا",
      "displayName-count-zero": "دولار غيانا",
      "displayName-count-one": "دولار غيانا",
      "displayName-count-two": "دولار غيانا",
      "displayName-count-few": "دولار غيانا",
      "displayName-count-many": "دولار غيانا",
      "displayName-count-other": "دولار غيانا",
      "symbol": "GYD",
      "symbol-alt-narrow": "GY$"
    },
    "HKD": {
      "displayName": "دولار هونغ كونغ",
      "displayName-count-zero": "دولار هونغ كونغ",
      "displayName-count-one": "دولار هونغ كونغ",
      "displayName-count-two": "دولار هونغ كونغ",

```

```

    "displayName-count-few": "دولار هونغ كونغ",
    "displayName-count-many": "دولار هونغ كونغ",
    "displayName-count-other": "دولار هونغ كونغ",
    "symbol": "HK$",
    "symbol-alt-narrow": "HK$"
  },
  "HNL": {
    "displayName": "ليمبيرا هندوراس",
    "displayName-count-zero": "ليمبيرا هندوراس",
    "displayName-count-one": "ليمبيرا هندوراس",
    "displayName-count-two": "ليمبيرا هندوراس",
    "displayName-count-few": "ليمبيرا هندوراس",
    "displayName-count-many": "ليمبيرا هندوراس",
    "displayName-count-other": "ليمبيرا هندوراس",
    "symbol": "HNL",
    "symbol-alt-narrow": "L"
  },
  "HRD": {
    "displayName": "دينار كرواتي",
    "symbol": "HRD"
  },
  "HRK": {
    "displayName": "كونا كرواتي",
    "displayName-count-zero": "كونا كرواتي",
    "displayName-count-one": "كونا كرواتي",
    "displayName-count-two": "كونا كرواتي",
    "displayName-count-few": "كونا كرواتي",
    "displayName-count-many": "كونا كرواتي",
    "displayName-count-other": "كونا كرواتي",
    "symbol": "HRK",
    "symbol-alt-narrow": "kn"
  },
  "HTG": {
    "displayName": "جوردي هايتي",
    "displayName-count-zero": "جوردي هايتي",
    "displayName-count-one": "جوردي هايتي",
    "displayName-count-two": "جوردي هايتي",
    "displayName-count-few": "جوردي هايتي",
    "displayName-count-many": "جوردي هايتي",
    "displayName-count-other": "جوردي هايتي",
    "symbol": "HTG"
  },
  "HUF": {
    "displayName": "فورينت مجري",
    "displayName-count-zero": "فورينت مجري",
    "displayName-count-one": "فورينت مجري",
    "displayName-count-two": "فورينت مجري",
    "displayName-count-few": "فورينت مجري",
    "displayName-count-many": "فورينت مجري",
    "displayName-count-other": "فورينت مجري",
    "symbol": "HUF",
    "symbol-alt-narrow": "Ft"
  },
  "IDR": {
    "displayName": "روبية إندونيسية",
    "displayName-count-zero": "روبية إندونيسية",
    "displayName-count-one": "روبية إندونيسية",

```

```

    "displayName-count-two": "روبية إندونيسية",
    "displayName-count-few": "روبية إندونيسية",
    "displayName-count-many": "روبية إندونيسية",
    "displayName-count-other": "روبية إندونيسية",
    "symbol": "IDR",
    "symbol-alt-narrow": "Rp"
  },
  "IEP": {
    "displayName": "جنيه إيرلندي",
    "symbol": "IEP"
  },
  "ILP": {
    "displayName": "جنيه إسرائيلي",
    "symbol": "ILP"
  },
  "ILS": {
    "displayName": "شيكل إسرائيلي جديد",
    "displayName-count-zero": "شيكل إسرائيلي جديد",
    "displayName-count-one": "شيكل إسرائيلي جديد",
    "displayName-count-two": "شيكل إسرائيلي جديد",
    "displayName-count-few": "شيكل إسرائيلي جديد",
    "displayName-count-many": "شيكل إسرائيلي جديد",
    "displayName-count-other": "شيكل إسرائيلي جديد",
    "symbol": "₪",
    "symbol-alt-narrow": "₪"
  },
  "INR": {
    "displayName": "روبية هندي",
    "displayName-count-zero": "روبية هندي",
    "displayName-count-one": "روبية هندي",
    "displayName-count-two": "روبية هندي",
    "displayName-count-few": "روبية هندي",
    "displayName-count-many": "روبية هندي",
    "displayName-count-other": "روبية هندي",
    "symbol": "₹",
    "symbol-alt-narrow": "₹"
  },
  "IQD": {
    "displayName": "دينار عراقي",
    "displayName-count-zero": "دينار عراقي",
    "displayName-count-one": "دينار عراقي",
    "displayName-count-two": "دينار عراقي",
    "displayName-count-few": "دينار عراقي",
    "displayName-count-many": "دينار عراقي",
    "displayName-count-other": "دينار عراقي",
    "symbol": "د.ع."
  },
  "IRR": {
    "displayName": "ريال إيراني",
    "displayName-count-zero": "ريال إيراني",
    "displayName-count-one": "ريال إيراني",
    "displayName-count-two": "ريال إيراني",
    "displayName-count-few": "ريال إيراني",
    "displayName-count-many": "ريال إيراني",
    "displayName-count-other": "ريال إيراني",
    "symbol": "ر.إ."
  },
}

```

```

"ISK": {
  "displayName": "كرونة آيسلندي",
  "displayName-count-zero": "كرونة آيسلندي",
  "displayName-count-one": "كرونة آيسلندي",
  "displayName-count-two": "كرونة آيسلندي",
  "displayName-count-few": "كرونة آيسلندي",
  "displayName-count-many": "كرونة آيسلندي",
  "displayName-count-other": "كرونة آيسلندي",
  "symbol": "ISK",
  "symbol-alt-narrow": "kr"
},
"ITL": {
  "displayName": "ليرة إيطالية",
  "symbol": "ITL"
},
"JMD": {
  "displayName": "دولار جامايكي",
  "displayName-count-zero": "دولار جامايكي",
  "displayName-count-one": "دولار جامايكي",
  "displayName-count-two": "دولار جامايكي",
  "displayName-count-few": "دولار جامايكي",
  "displayName-count-many": "دولار جامايكي",
  "displayName-count-other": "دولار جامايكي",
  "symbol": "JMD",
  "symbol-alt-narrow": "JM$"
},
"JOD": {
  "displayName": "دينار أردني",
  "displayName-count-zero": "دينار أردني",
  "displayName-count-one": "دينار أردني",
  "displayName-count-two": "دينار أردني",
  "displayName-count-few": "دينار أردني",
  "displayName-count-many": "دينار أردني",
  "displayName-count-other": "دينار أردني",
  "symbol": "د.أ."
},
"JPY": {
  "displayName": "ين ياباني",
  "displayName-count-zero": "ين ياباني",
  "displayName-count-one": "ين ياباني",
  "displayName-count-two": "ين ياباني",
  "displayName-count-few": "ين ياباني",
  "displayName-count-many": "ين ياباني",
  "displayName-count-other": "ين ياباني",
  "symbol": "JP¥",
  "symbol-alt-narrow": "JP¥"
},
"KES": {
  "displayName": "شلن كيني",
  "displayName-count-zero": "شلن كيني",
  "displayName-count-one": "شلن كيني",
  "displayName-count-two": "شلن كيني",
  "displayName-count-few": "شلن كيني",
  "displayName-count-many": "شلن كيني",
  "displayName-count-other": "شلن كيني",
  "symbol": "KES"
},

```

```

"KGS": {
  "displayName": "سوم قيرغستاني",
  "displayName-count-zero": "سوم قيرغستاني",
  "displayName-count-one": "سوم قيرغستاني",
  "displayName-count-two": "سوم قيرغستاني",
  "displayName-count-few": "سوم قيرغستاني",
  "displayName-count-many": "سوم قيرغستاني",
  "displayName-count-other": "سوم قيرغستاني",
  "symbol": "KGS"
},
"KHR": {
  "displayName": "ريال كمبودي",
  "displayName-count-zero": "ريال كمبودي",
  "displayName-count-one": "ريال كمبودي",
  "displayName-count-two": "ريال كمبودي",
  "displayName-count-few": "ريال كمبودي",
  "displayName-count-many": "ريال كمبودي",
  "displayName-count-other": "ريال كمبودي",
  "symbol": "KHR",
  "symbol-alt-narrow": "៛"
},
"KMF": {
  "displayName": "فرنك جزر القمر",
  "displayName-count-zero": "فرنك جزر القمر",
  "displayName-count-one": "فرنك جزر القمر",
  "displayName-count-two": "فرنك جزر القمر",
  "displayName-count-few": "فرنك جزر القمر",
  "displayName-count-many": "فرنك جزر القمر",
  "displayName-count-other": "فرنك جزر القمر",
  "symbol": "ف.ج.ق.",
  "symbol-alt-narrow": "CF"
},
"KPW": {
  "displayName": "وون كوريا الشمالية",
  "displayName-count-zero": "وون كوريا الشمالية",
  "displayName-count-one": "وون كوريا الشمالية",
  "displayName-count-two": "وون كوريا الشمالية",
  "displayName-count-few": "وون كوريا الشمالية",
  "displayName-count-many": "وون كوريا الشمالية",
  "displayName-count-other": "وون كوريا الشمالية",
  "symbol": "KPW",
  "symbol-alt-narrow": "₩"
},
"KRH": {
  "displayName": "KRH",
  "symbol": "KRH"
},
"KRO": {
  "displayName": "KRO",
  "symbol": "KRO"
},
"KRW": {
  "displayName": "وون كوريا الجنوبية",
  "displayName-count-zero": "وون كوريا الجنوبية",
  "displayName-count-one": "وون كوريا الجنوبية",
  "displayName-count-two": "وون كوريا الجنوبية",

```



```

    "displayName-count-few": "وون كوريا الجنوبية",
    "displayName-count-many": "وون كوريا الجنوبية",
    "displayName-count-other": "وون كوريا الجنوبية",
    "symbol": "₩",
    "symbol-alt-narrow": "₩"
  },
  "KWD": {
    "displayName": "دينار كويتي",
    "displayName-count-zero": "دينار كويتي",
    "displayName-count-one": "دينار كويتي",
    "displayName-count-two": "دينار كويتي",
    "displayName-count-few": "دينار كويتي",
    "displayName-count-many": "دينار كويتي",
    "displayName-count-other": "دينار كويتي",
    "symbol": "د.ك."
  },
  "KYD": {
    "displayName": "دولار جزر كيمن",
    "displayName-count-zero": "دولار جزر كيمن",
    "displayName-count-one": "دولار جزر كيمن",
    "displayName-count-two": "دولار جزر كيمن",
    "displayName-count-few": "دولار جزر كيمن",
    "displayName-count-many": "دولار جزر كيمن",
    "displayName-count-other": "دولار جزر كيمن",
    "symbol": "KYD",
    "symbol-alt-narrow": "KY$"
  },
  "KZT": {
    "displayName": "تينغ كازاخستاني",
    "displayName-count-zero": "تينغ كازاخستاني",
    "displayName-count-one": "تينغ كازاخستاني",
    "displayName-count-two": "تينغ كازاخستاني",
    "displayName-count-few": "تينغ كازاخستاني",
    "displayName-count-many": "تينغ كازاخستاني",
    "displayName-count-other": "تينغ كازاخستاني",
    "symbol": "KZT",
    "symbol-alt-narrow": "₸"
  },
  "LAK": {
    "displayName": "كيب لاوسي",
    "displayName-count-zero": "كيب لاوسي",
    "displayName-count-one": "كيب لاوسي",
    "displayName-count-two": "كيب لاوسي",
    "displayName-count-few": "كيب لاوسي",
    "displayName-count-many": "كيب لاوسي",
    "displayName-count-other": "كيب لاوسي",
    "symbol": "LAK",
    "symbol-alt-narrow": "₭"
  },
  "LBP": {
    "displayName": "جنيه لبناني",
    "displayName-count-zero": "جنيه لبناني",
    "displayName-count-one": "جنيه لبناني",
    "displayName-count-two": "جنيه لبناني",
    "displayName-count-few": "جنيه لبناني",
    "displayName-count-many": "جنيه لبناني",
    "displayName-count-other": "جنيه لبناني",

```

```

    "symbol": ".J.J",
    "symbol-alt-narrow": "Lℓ"
  },
  "LKR": {
    "displayName": "روبية سريلانكية",
    "displayName-count-zero": "روبية سريلانكية",
    "displayName-count-one": "روبية سريلانكية",
    "displayName-count-two": "روبية سريلانكية",
    "displayName-count-few": "روبية سريلانكية",
    "displayName-count-many": "روبية سريلانكية",
    "displayName-count-other": "روبية سريلانكية",
    "symbol": "LKR",
    "symbol-alt-narrow": "Rs"
  },
  "LRD": {
    "displayName": "دولار ليبيري",
    "displayName-count-zero": "دولار ليبيري",
    "displayName-count-one": "دولار ليبيري",
    "displayName-count-two": "دولاران ليبيريان",
    "displayName-count-few": "دولارات ليبيرية",
    "displayName-count-many": "دولارًا ليبيريًا",
    "displayName-count-other": "دولار ليبيري",
    "symbol": "LRD",
    "symbol-alt-narrow": "$"
  },
  "LSL": {
    "displayName": "لوتي ليسوتو",
    "symbol": "LSL"
  },
  "LTL": {
    "displayName": "ليتة ليتوانية",
    "displayName-count-zero": "ليتة ليتوانية",
    "displayName-count-one": "ليتة ليتوانية",
    "displayName-count-two": "ليتة ليتوانية",
    "displayName-count-few": "ليتة ليتوانية",
    "displayName-count-many": "ليتة ليتوانية",
    "displayName-count-other": "ليتة ليتوانية",
    "symbol": "LTL",
    "symbol-alt-narrow": "Lt"
  },
  "LTT": {
    "displayName": "تالوناس ليتواني",
    "symbol": "LTT"
  },
  "LUC": {
    "displayName": "فرنك لوكسمبرج قابل للتحويل",
    "symbol": "LUC"
  },
  "LUF": {
    "displayName": "فرنك لوكسمبرج",
    "symbol": "LUF"
  },
  "LUL": {
    "displayName": "فرنك لوكسمبرج المالي",
    "symbol": "LUL"
  },
  "LVL": {

```

```

    "displayName": "لاتس لاتفيا",
    "displayName-count-zero": "لاتس لاتفي",
    "displayName-count-one": "لاتس لاتفي",
    "displayName-count-two": "لاتس لاتفي",
    "displayName-count-few": "لاتس لاتفي",
    "displayName-count-many": "لاتس لاتفي",
    "displayName-count-other": "لاتس لاتفي",
    "symbol": "LVL",
    "symbol-alt-narrow": "Ls"
  },
  "LVR": {
    "displayName": "روبل لاتفيا",
    "symbol": "LVR"
  },
  "LYD": {
    "displayName": "دينار ليبي",
    "displayName-count-zero": "دينار ليبي",
    "displayName-count-one": "دينار ليبي",
    "displayName-count-two": "ديناران ليبيان",
    "displayName-count-few": "دينارات ليبية",
    "displayName-count-many": "دينارًا ليبيا",
    "displayName-count-other": "دينار ليبي",
    "symbol": "د.ل."
  },
  "MAD": {
    "displayName": "درهم مغربي",
    "displayName-count-zero": "درهم مغربي",
    "displayName-count-one": "درهم مغربي",
    "displayName-count-two": "درهمان مغربيان",
    "displayName-count-few": "دراهم مغربية",
    "displayName-count-many": "درهمًا مغربيًا",
    "displayName-count-other": "درهم مغربي",
    "symbol": "د.م."
  },
  "MAF": {
    "displayName": "فرنك مغربي",
    "symbol": "MAF"
  },
  "MCF": {
    "displayName": "MCF",
    "symbol": "MCF"
  },
  "MDC": {
    "displayName": "MDC",
    "symbol": "MDC"
  },
  "MDL": {
    "displayName": "ليو مولدوفي",
    "displayName-count-zero": "ليو مولدوفي",
    "displayName-count-one": "ليو مولدوفي",
    "displayName-count-two": "ليو مولدوفي",
    "displayName-count-few": "ليو مولدوفي",
    "displayName-count-many": "ليو مولدوفي",
    "displayName-count-other": "ليو مولدوفي",
    "symbol": "MDL"
  },
  "MGA": {

```

```

    "displayName": "أرياري مدغشقر",
    "displayName-count-zero": "أرياري مدغشقر",
    "displayName-count-one": "أرياري مدغشقر",
    "displayName-count-two": "أرياري مدغشقر",
    "displayName-count-few": "أرياري مدغشقر",
    "displayName-count-many": "أرياري مدغشقر",
    "displayName-count-other": "أرياري مدغشقر",
    "symbol": "MGA",
    "symbol-alt-narrow": "Ar"
  },
  "MGF": {
    "displayName": "فرنك مدغشقر",
    "symbol": "MGF"
  },
  "MKD": {
    "displayName": "دينار مقدوني",
    "displayName-count-zero": "دينار مقدوني",
    "displayName-count-one": "دينار مقدوني",
    "displayName-count-two": "ديناران مقدونيان",
    "displayName-count-few": "دينارات مقدونية",
    "displayName-count-many": "دينارًا مقدونيًا",
    "displayName-count-other": "دينار مقدوني",
    "symbol": "MKD"
  },
  "MKN": {
    "displayName": "MKN",
    "symbol": "MKN"
  },
  "MLF": {
    "displayName": "فرنك مالي",
    "symbol": "MLF"
  },
  "MMK": {
    "displayName": "كيات ميانمار",
    "displayName-count-zero": "كيات ميانمار",
    "displayName-count-one": "كيات ميانمار",
    "displayName-count-two": "كيات ميانمار",
    "displayName-count-few": "كيات ميانمار",
    "displayName-count-many": "كيات ميانمار",
    "displayName-count-other": "كيات ميانمار",
    "symbol": "MMK",
    "symbol-alt-narrow": "K"
  },
  "MNT": {
    "displayName": "توغروغ منغولي",
    "displayName-count-zero": "توغروغ منغولي",
    "displayName-count-one": "توغروغ منغولي",
    "displayName-count-two": "توغروغ منغولي",
    "displayName-count-few": "توغروغ منغولي",
    "displayName-count-many": "توغروغ منغولي",
    "displayName-count-other": "توغروغ منغولي",
    "symbol": "MNT",
    "symbol-alt-narrow": "₮"
  },
  "MOP": {
    "displayName": "باتاكا ماكاوي",
    "displayName-count-zero": "باتاكا ماكاوي",

```

```

    "displayName-count-one": "باتاكا ماکاوي",
    "displayName-count-two": "باتاكا ماکاوي",
    "displayName-count-few": "باتاكا ماکاوي",
    "displayName-count-many": "باتاكا ماکاوي",
    "displayName-count-other": "باتاكا ماکاوي",
    "symbol": "MOP"
  },
  "MRO": {
    "displayName": "أوقية موريتانية",
    "displayName-count-zero": "أوقية موريتانية",
    "displayName-count-one": "أوقية موريتانية",
    "displayName-count-two": "أوقية موريتانية",
    "displayName-count-few": "أوقية موريتانية",
    "displayName-count-many": "أوقية موريتانية",
    "displayName-count-other": "أوقية موريتانية",
    "symbol": "أ.م."
  },
  "MTL": {
    "displayName": "ليرة مالطية",
    "symbol": "MTL"
  },
  "MTP": {
    "displayName": "جنيه مالطي",
    "symbol": "MTP"
  },
  "MUR": {
    "displayName": "روبية موريشيوسية",
    "displayName-count-zero": "روبية موريشيوسية",
    "displayName-count-one": "روبية موريشيوسية",
    "displayName-count-two": "روبية موريشيوسية",
    "displayName-count-few": "روبية موريشيوسية",
    "displayName-count-many": "روبية موريشيوسية",
    "displayName-count-other": "روبية موريشيوسية",
    "symbol": "MUR",
    "symbol-alt-narrow": "Rs"
  },
  "MVR": {
    "displayName": "روفيه جزر المالديف",
    "displayName-count-zero": "روفيه جزر المالديف",
    "displayName-count-one": "روفيه جزر المالديف",
    "displayName-count-two": "روفيه جزر المالديف",
    "displayName-count-few": "روفيه جزر المالديف",
    "displayName-count-many": "روفيه جزر المالديف",
    "displayName-count-other": "روفيه جزر المالديف",
    "symbol": "MVR"
  },
  "MWK": {
    "displayName": "كواشا مالاوي",
    "displayName-count-zero": "كواشا مالاوي",
    "displayName-count-one": "كواشا مالاوي",
    "displayName-count-two": "كواشا مالاوي",
    "displayName-count-few": "كواشا مالاوي",
    "displayName-count-many": "كواشا مالاوي",
    "displayName-count-other": "كواشا مالاوي",
    "symbol": "MWK"
  },
  "MXN": {

```

```

      "displayName": "بيزو مكسيكي",
      "displayName-count-zero": "بيزو مكسيكي",
      "displayName-count-one": "بيزو مكسيكي",
      "displayName-count-two": "بيزو مكسيكي",
      "displayName-count-few": "بيزو مكسيكي",
      "displayName-count-many": "بيزو مكسيكي",
      "displayName-count-other": "بيزو مكسيكي",
      "symbol": "MX$",
      "symbol-alt-narrow": "MX$"
    },
    "MXP": {
      "displayName": "بيزو فضي مكسيكي - 1992-1861",
      "symbol": "MXP"
    },
    "MXV": {
      "displayName": "MXV",
      "symbol": "MXV"
    },
    "MYR": {
      "displayName": "رينغيت ماليزي",
      "displayName-count-zero": "رينغيت ماليزي",
      "displayName-count-one": "رينغيت ماليزي",
      "displayName-count-two": "رينغيت ماليزي",
      "displayName-count-few": "رينغيت ماليزي",
      "displayName-count-many": "رينغيت ماليزي",
      "displayName-count-other": "رينغيت ماليزي",
      "symbol": "MYR",
      "symbol-alt-narrow": "RM"
    },
    "MZE": {
      "displayName": "اسكود موزمبيقي",
      "symbol": "MZE"
    },
    "MZM": {
      "displayName": "MZM",
      "symbol": "MZM"
    },
    "MZN": {
      "displayName": "متكال موزمبيقي",
      "displayName-count-zero": "متكال موزمبيقي",
      "displayName-count-one": "متكال موزمبيقي",
      "displayName-count-two": "متكال موزمبيقي",
      "displayName-count-few": "متكال موزمبيقي",
      "displayName-count-many": "متكال موزمبيقي",
      "displayName-count-other": "متكال موزمبيقي",
      "symbol": "MZN"
    },
    "NAD": {
      "displayName": "دولار ناميبي",
      "displayName-count-zero": "دولار ناميبي",
      "displayName-count-one": "دولار ناميبي",
      "displayName-count-two": "دولار ناميبي",
      "displayName-count-few": "دولار ناميبي",
      "displayName-count-many": "دولار ناميبي",
      "displayName-count-other": "دولار ناميبي",
      "symbol": "NAD",
      "symbol-alt-narrow": "$"
    }
  }

```

```

    },
    "NGN": {
      "displayName": "نايرا نيجيري",
      "displayName-count-zero": "نايرا نيجيري",
      "displayName-count-one": "نايرا نيجيري",
      "displayName-count-two": "نايرا نيجيري",
      "displayName-count-few": "نايرا نيجيري",
      "displayName-count-many": "نايرا نيجيري",
      "displayName-count-other": "نايرا نيجيري",
      "symbol": "NGN",
      "symbol-alt-narrow": "₦"
    },
    "NIC": {
      "displayName": "كوردوبه نيكاراغوا",
      "symbol": "NIC"
    },
    "NIO": {
      "displayName": "قرطبة نيكاراغوا",
      "displayName-count-zero": "قرطبة نيكاراغوا",
      "displayName-count-one": "قرطبة نيكاراغوا",
      "displayName-count-two": "قرطبة نيكاراغوا",
      "displayName-count-few": "قرطبة نيكاراغوا",
      "displayName-count-many": "قرطبة نيكاراغوا",
      "displayName-count-other": "قرطبة نيكاراغوا",
      "symbol": "NIO",
      "symbol-alt-narrow": "C$"
    },
    "NLG": {
      "displayName": "جلدر هولندي",
      "symbol": "NLG"
    },
    "NOK": {
      "displayName": "كرونه نرويجية",
      "displayName-count-zero": "كرونه نرويجية",
      "displayName-count-one": "كرونه نرويجية",
      "displayName-count-two": "كرونه نرويجية",
      "displayName-count-few": "كرونه نرويجية",
      "displayName-count-many": "كرونه نرويجية",
      "displayName-count-other": "كرونه نرويجية",
      "symbol": "NOK",
      "symbol-alt-narrow": "kr"
    },
    "NPR": {
      "displayName": "روبية نيبالي",
      "displayName-count-zero": "روبية نيبالي",
      "displayName-count-one": "روبية نيبالي",
      "displayName-count-two": "روبية نيبالي",
      "displayName-count-few": "روبية نيبالي",
      "displayName-count-many": "روبية نيبالي",
      "displayName-count-other": "روبية نيبالي",
      "symbol": "NPR",
      "symbol-alt-narrow": "Rs"
    },
    "NZD": {
      "displayName": "دولار نيوزيلندي",
      "displayName-count-zero": "دولار نيوزيلندي",
      "displayName-count-one": "دولار نيوزيلندي",

```

```

    "displayName-count-two": "دولار نيوزيلندي",
    "displayName-count-few": "دولار نيوزيلندي",
    "displayName-count-many": "دولار نيوزيلندي",
    "displayName-count-other": "دولار نيوزيلندي",
    "symbol": "NZ$",
    "symbol-alt-narrow": "NZ$"
  },
  "OMR": {
    "displayName": "ر.ع.ع",
    "displayName-count-zero": "ر.ع.ع",
    "displayName-count-one": "ر.ع.ع",
    "displayName-count-two": "ر.ع.ع",
    "displayName-count-few": "ر.ع.ع",
    "displayName-count-many": "ر.ع.ع",
    "displayName-count-other": "ر.ع.ع",
    "symbol": "ر.ع.ع"
  },
  "PAB": {
    "displayName": "بالبوا بنمي",
    "displayName-count-zero": "بالبوا بنمي",
    "displayName-count-one": "بالبوا بنمي",
    "displayName-count-two": "بالبوا بنمي",
    "displayName-count-few": "بالبوا بنمي",
    "displayName-count-many": "بالبوا بنمي",
    "displayName-count-other": "بالبوا بنمي",
    "symbol": "PAB"
  },
  "PEI": {
    "displayName": "PEI",
    "symbol": "PEI"
  },
  "PEN": {
    "displayName": "سول جديد البيرو",
    "displayName-count-zero": "سول جديد البيرو",
    "displayName-count-one": "سول جديد البيرو",
    "displayName-count-two": "سول جديد البيرو",
    "displayName-count-few": "سول جديد البيرو",
    "displayName-count-many": "سول جديد البيرو",
    "displayName-count-other": "سول جديد البيرو",
    "symbol": "PEN"
  },
  "PES": {
    "displayName": "PES",
    "symbol": "PES"
  },
  "PGK": {
    "displayName": "كينيا بابوا غينيا الجديدة",
    "displayName-count-zero": "كينيا بابوا غينيا الجديدة",
    "displayName-count-one": "كينيا بابوا غينيا الجديدة",
    "displayName-count-two": "كينيا بابوا غينيا الجديدة",
    "displayName-count-few": "كينيا بابوا غينيا الجديدة",
    "displayName-count-many": "كينيا بابوا غينيا الجديدة",
    "displayName-count-other": "كينيا بابوا غينيا الجديدة",
    "symbol": "PGK"
  },
  "PHP": {
    "displayName": "بيزو فلپيني",

```



```

    "displayName-count-zero": "بیزو فلپینی",
    "displayName-count-one": "بیزو فلپینی",
    "displayName-count-two": "بیزو فلپینی",
    "displayName-count-few": "بیزو فلپینی",
    "displayName-count-many": "بیزو فلپینی",
    "displayName-count-other": "بیزو فلپینی",
    "symbol": "PHP",
    "symbol-alt-narrow": "₱"
  },
  "PKR": {
    "displayName": "روبیۂ پاکستانی",
    "displayName-count-zero": "روبیۂ پاکستانی",
    "displayName-count-one": "روبیۂ پاکستانی",
    "displayName-count-two": "روبیۂ پاکستانی",
    "displayName-count-few": "روبیۂ پاکستانی",
    "displayName-count-many": "روبیۂ پاکستانی",
    "displayName-count-other": "روبیۂ پاکستانی",
    "symbol": "PKR",
    "symbol-alt-narrow": "Rs"
  },
  "PLN": {
    "displayName": "زلوٹی بولندی",
    "displayName-count-zero": "زلوٹی بولندی",
    "displayName-count-one": "زلوٹی بولندی",
    "displayName-count-two": "زلوٹی بولندی",
    "displayName-count-few": "زلوٹی بولندی",
    "displayName-count-many": "زلوٹی بولندی",
    "displayName-count-other": "زلوٹی بولندی",
    "symbol": "PLN",
    "symbol-alt-narrow": "zł"
  },
  "PLZ": {
    "displayName": "زلوٹی بولندی - 1950-1995",
    "symbol": "PLZ"
  },
  "PTE": {
    "displayName": "اسکود برتغالی",
    "symbol": "PTE"
  },
  "PYG": {
    "displayName": "جوارانی باراجوای",
    "displayName-count-zero": "جوارانی باراجوای",
    "displayName-count-one": "جوارانی باراجوای",
    "displayName-count-two": "جوارانی باراجوای",
    "displayName-count-few": "جوارانی باراجوای",
    "displayName-count-many": "جوارانی باراجوای",
    "displayName-count-other": "جوارانی باراجوای",
    "symbol": "PYG",
    "symbol-alt-narrow": "₧"
  },
  "QAR": {
    "displayName": "وق قطري",
    "displayName-count-zero": "وق قطري",
    "displayName-count-one": "وق قطري",
    "displayName-count-two": "وق قطري",
    "displayName-count-few": "وق قطري",
    "displayName-count-many": "وق قطري",

```

```

        "displayName-count-other": "ر.ق. قطري",
        "symbol": "ر.ق."
    },
    "RHD": {
        "displayName": "دولار روديسي",
        "symbol": "RHD"
    },
    "ROL": {
        "displayName": "ليو روماني قديم",
        "symbol": "ROL"
    },
    "RON": {
        "displayName": "ليو روماني",
        "displayName-count-zero": "ليو روماني",
        "displayName-count-one": "ليو روماني",
        "displayName-count-two": "ليو روماني",
        "displayName-count-few": "ليو روماني",
        "displayName-count-many": "ليو روماني",
        "displayName-count-other": "ليو روماني",
        "symbol": "RON",
        "symbol-alt-narrow": "lei"
    },
    "RSD": {
        "displayName": "دينار صربي",
        "displayName-count-zero": "دينار صربي",
        "displayName-count-one": "دينار صربي",
        "displayName-count-two": "ديناران صربيان",
        "displayName-count-few": "دينارات صربية",
        "displayName-count-many": "دينارًا صربيًا",
        "displayName-count-other": "دينار صربي",
        "symbol": "RSD"
    },
    "RUB": {
        "displayName": "روبل روسي",
        "displayName-count-zero": "روبل روسي",
        "displayName-count-one": "روبل روسي",
        "displayName-count-two": "روبل روسي",
        "displayName-count-few": "روبل روسي",
        "displayName-count-many": "روبل روسي",
        "displayName-count-other": "روبل روسي",
        "symbol": "RUB",
        "symbol-alt-narrow": "₽"
    },
    "RUR": {
        "displayName": "1998-1991 - روبل روسي",
        "symbol": "RUR",
        "symbol-alt-narrow": "p."
    },
    "RWF": {
        "displayName": "فرنك رواندي",
        "displayName-count-zero": "فرنك رواندي",
        "displayName-count-one": "فرنك رواندي",
        "displayName-count-two": "فرنك رواندي",
        "displayName-count-few": "فرنك رواندي",
        "displayName-count-many": "فرنك رواندي",
        "displayName-count-other": "فرنك رواندي",
        "symbol": "RWF",

```

```

    "symbol-alt-narrow": "RF"
  },
  "SAR": {
    "displayName": "ريال سعودي",
    "displayName-count-zero": "ريال سعودي",
    "displayName-count-one": "ريال سعودي",
    "displayName-count-two": "ريال سعودي",
    "displayName-count-few": "ريال سعودي",
    "displayName-count-many": "ريال سعودي",
    "displayName-count-other": "ريال سعودي",
    "symbol": "ر.س."
  },
  "SBD": {
    "displayName": "دولار جزر سليمان",
    "displayName-count-zero": "دولار جزر سليمان",
    "displayName-count-one": "دولار جزر سليمان",
    "displayName-count-two": "دولار جزر سليمان",
    "displayName-count-few": "دولار جزر سليمان",
    "displayName-count-many": "دولار جزر سليمان",
    "displayName-count-other": "دولار جزر سليمان",
    "symbol": "SBD",
    "symbol-alt-narrow": "SB$"
  },
  "SCR": {
    "displayName": "روبية سيشيلية",
    "displayName-count-zero": "روبية سيشيلية",
    "displayName-count-one": "روبية سيشيلية",
    "displayName-count-two": "روبية سيشيلية",
    "displayName-count-few": "روبية سيشيلية",
    "displayName-count-many": "روبية سيشيلية",
    "displayName-count-other": "روبية سيشيلية",
    "symbol": "SCR"
  },
  "SDD": {
    "displayName": "دينار سوداني",
    "symbol": "د.س."
  },
  "SDG": {
    "displayName": "جنيه سوداني",
    "displayName-count-zero": "جنيه سوداني",
    "displayName-count-one": "جنيه سوداني",
    "displayName-count-two": "جنيه سوداني",
    "displayName-count-few": "جنيهات سودانية",
    "displayName-count-many": "جنيهًا سودانيًا",
    "displayName-count-other": "جنيه سوداني",
    "symbol": "ج.س."
  },
  "SDP": {
    "displayName": "جنيه سوداني قديم",
    "symbol": "SDP"
  },
  "SEK": {
    "displayName": "كرونة سويدية",
    "displayName-count-zero": "كرونة سويدية",
    "displayName-count-one": "كرونة سويدية",
    "displayName-count-two": "كرونة سويدية",
    "displayName-count-few": "كرونة سويدية",

```

```

    "displayName-count-many": "كرونة سويدية",
    "displayName-count-other": "كرونة سويدية",
    "symbol": "SEK",
    "symbol-alt-narrow": "kr"
  },
  "SGD": {
    "displayName": "دولار سنغافوري",
    "displayName-count-zero": "دولار سنغافوري",
    "displayName-count-one": "دولار سنغافوري",
    "displayName-count-two": "دولار سنغافوري",
    "displayName-count-few": "دولار سنغافوري",
    "displayName-count-many": "دولار سنغافوري",
    "displayName-count-other": "دولار سنغافوري",
    "symbol": "SGD",
    "symbol-alt-narrow": "$"
  },
  "SHP": {
    "displayName": "جنيه سانت هيلين",
    "displayName-count-zero": "جنيه سانت هيلين",
    "displayName-count-one": "جنيه سانت هيلين",
    "displayName-count-two": "جنيه سانت هيلين",
    "displayName-count-few": "جنيه سانت هيلين",
    "displayName-count-many": "جنيه سانت هيلين",
    "displayName-count-other": "جنيه سانت هيلين",
    "symbol": "SHP",
    "symbol-alt-narrow": "£"
  },
  "SIT": {
    "displayName": "تولار سلوفيني",
    "symbol": "SIT"
  },
  "SKK": {
    "displayName": "كرونة سلوفاكية",
    "symbol": "SKK"
  },
  "SLL": {
    "displayName": "ليون سيراليوني",
    "displayName-count-zero": "ليون سيراليوني",
    "displayName-count-one": "ليون سيراليوني",
    "displayName-count-two": "ليون سيراليوني",
    "displayName-count-few": "ليون سيراليوني",
    "displayName-count-many": "ليون سيراليوني",
    "displayName-count-other": "ليون سيراليوني",
    "symbol": "SLL"
  },
  "SOS": {
    "displayName": "شلن صومالي",
    "displayName-count-zero": "شلن صومالي",
    "displayName-count-one": "شلن صومالي",
    "displayName-count-two": "شلن صومالي",
    "displayName-count-few": "شلن صومالي",
    "displayName-count-many": "شلن صومالي",
    "displayName-count-other": "شلن صومالي",
    "symbol": "SOS"
  },
  "SRD": {
    "displayName": "دولار سورينامي",

```

```

    "displayName-count-zero": "دولار سورينامي",
    "displayName-count-one": "دولار سورينامي",
    "displayName-count-two": "دولار سورينامي",
    "displayName-count-few": "دولار سورينامي",
    "displayName-count-many": "دولار سورينامي",
    "displayName-count-other": "دولار سورينامي",
    "symbol": "SRD",
    "symbol-alt-narrow": "SR$"
  },
  "SRG": {
    "displayName": "جلدر سورينامي",
    "symbol": "SRG"
  },
  "SSP": {
    "displayName": "جنيه جنوب السودان",
    "displayName-count-zero": "جنيه جنوب السودان",
    "displayName-count-one": "جنيه جنوب السودان",
    "displayName-count-two": "جنيهان جنوب السودان",
    "displayName-count-few": "جنيهات جنوب السودان",
    "displayName-count-many": "جنيها جنوب السودان",
    "displayName-count-other": "جنيه جنوب السودان",
    "symbol": "SSP",
    "symbol-alt-narrow": "£"
  },
  "STD": {
    "displayName": "دوبرا ساو تومي وبرينسيبي",
    "displayName-count-zero": "دوبرا ساو تومي وبرينسيبي",
    "displayName-count-one": "دوبرا ساو تومي وبرينسيبي",
    "displayName-count-two": "دوبرا ساو تومي وبرينسيبي",
    "displayName-count-few": "دوبرا ساو تومي وبرينسيبي",
    "displayName-count-many": "دوبرا ساو تومي وبرينسيبي",
    "displayName-count-other": "دوبرا ساو تومي وبرينسيبي",
    "symbol": "STD",
    "symbol-alt-narrow": "Db"
  },
  "SUR": {
    "displayName": "روبل سوفيتي",
    "symbol": "SUR"
  },
  "SVC": {
    "displayName": "كولون سلفادوري",
    "symbol": "SVC"
  },
  "SYP": {
    "displayName": "ليرة سورية",
    "displayName-count-zero": "ليرة سورية",
    "displayName-count-one": "ليرة سورية",
    "displayName-count-two": "ليرة سورية",
    "displayName-count-few": "ليرة سورية",
    "displayName-count-many": "ليرة سورية",
    "displayName-count-other": "ليرة سورية",
    "symbol": "ل.س.",
    "symbol-alt-narrow": "£"
  },
  "SZL": {
    "displayName": "ليلانجيني سوازيلندي",
    "displayName-count-zero": "ليلانجيني سوازيلندي",

```

```

    "displayName-count-one": "ليلانجيني سوازيلندي",
    "displayName-count-two": "ليلانجيني سوازيلندي",
    "displayName-count-few": "ليلانجيني سوازيلندي",
    "displayName-count-many": "ليلانجيني سوازيلندي",
    "displayName-count-other": "ليلانجيني سوازيلندي",
    "symbol": "SZL"
  },
  "THB": {
    "displayName": "باخت تايلاندي",
    "displayName-count-zero": "باخت تايلاندي",
    "displayName-count-one": "باخت تايلاندي",
    "displayName-count-two": "باخت تايلاندي",
    "displayName-count-few": "باخت تايلاندي",
    "displayName-count-many": "باخت تايلاندي",
    "displayName-count-other": "باخت تايلاندي",
    "symbol": "฿",
    "symbol-alt-narrow": "฿"
  },
  "TJR": {
    "displayName": "روبل طاجيكستاني",
    "symbol": "TJR"
  },
  "TJS": {
    "displayName": "سوموني طاجيكستاني",
    "displayName-count-zero": "سوموني طاجيكستاني",
    "displayName-count-one": "سوموني طاجيكستاني",
    "displayName-count-two": "سوموني طاجيكستاني",
    "displayName-count-few": "سوموني طاجيكستاني",
    "displayName-count-many": "سوموني طاجيكستاني",
    "displayName-count-other": "سوموني طاجيكستاني",
    "symbol": "TJS"
  },
  "TMM": {
    "displayName": "مانات تركمنستاني",
    "symbol": "TMM"
  },
  "TMT": {
    "displayName": "مانات تركمانستان",
    "displayName-count-zero": "مانات تركمانستان",
    "displayName-count-one": "مانات تركمانستان",
    "displayName-count-two": "مانات تركمانستان",
    "displayName-count-few": "مانات تركمانستان",
    "displayName-count-many": "مانات تركمانستان",
    "displayName-count-other": "مانات تركمانستان",
    "symbol": "TMT"
  },
  "TND": {
    "displayName": "دينار تونسي",
    "displayName-count-zero": "دينار تونسي",
    "displayName-count-one": "دينار تونسي",
    "displayName-count-two": "ديناران تونسيان",
    "displayName-count-few": "دينارات تونسية",
    "displayName-count-many": "دينارًا تونسيًا",
    "displayName-count-other": "دينار تونسي",
    "symbol": "د.ت."
  },
  "TOP": {

```

```

    "displayName": "بانغا تونغا",
    "displayName-count-zero": "بانغا تونغا",
    "displayName-count-one": "بانغا تونغا",
    "displayName-count-two": "بانغا تونغا",
    "displayName-count-few": "بانغا تونغا",
    "displayName-count-many": "بانغا تونغا",
    "displayName-count-other": "بانغا تونغا",
    "symbol": "TOP",
    "symbol-alt-narrow": "T$"
  },
  "TPE": {
    "displayName": "اسكود تيموري",
    "symbol": "TPE"
  },
  "TRL": {
    "displayName": "ليرة تركي",
    "symbol": "TRL"
  },
  "TRY": {
    "displayName": "ليرة تركية",
    "displayName-count-zero": "ليرة تركية",
    "displayName-count-one": "ليرة تركية",
    "displayName-count-two": "ليرة تركية",
    "displayName-count-few": "ليرة تركية",
    "displayName-count-many": "ليرة تركية",
    "displayName-count-other": "ليرة تركية",
    "symbol": "TRY",
    "symbol-alt-narrow": "₺",
    "symbol-alt-variant": "TL"
  },
  "TTD": {
    "displayName": "دولار ترينداد وتوباغو",
    "displayName-count-zero": "دولار ترينداد وتوباغو",
    "displayName-count-one": "دولار ترينداد وتوباغو",
    "displayName-count-two": "دولار ترينداد وتوباغو",
    "displayName-count-few": "دولار ترينداد وتوباغو",
    "displayName-count-many": "دولار ترينداد وتوباغو",
    "displayName-count-other": "دولار ترينداد وتوباغو",
    "symbol": "TTD",
    "symbol-alt-narrow": "TT$"
  },
  "TWD": {
    "displayName": "دولار تايواني",
    "displayName-count-zero": "دولار تايواني",
    "displayName-count-one": "دولار تايواني",
    "displayName-count-two": "دولار تايواني",
    "displayName-count-few": "دولار تايواني",
    "displayName-count-many": "دولار تايواني",
    "displayName-count-other": "دولار تايواني",
    "symbol": "NT$",
    "symbol-alt-narrow": "NT$"
  },
  "TZS": {
    "displayName": "شلن تنزاني",
    "displayName-count-zero": "شلن تنزاني",
    "displayName-count-one": "شلن تنزاني",
    "displayName-count-two": "شلن تنزاني",

```

```

    "displayName-count-few": "شلن تنزاني",
    "displayName-count-many": "شلن تنزاني",
    "displayName-count-other": "شلن تنزاني",
    "symbol": "TZS"
  },
  "UAH": {
    "displayName": "هريفنيا أوكراني",
    "displayName-count-zero": "هريفنيا أوكراني",
    "displayName-count-one": "هريفنيا أوكراني",
    "displayName-count-two": "هريفنيا أوكراني",
    "displayName-count-few": "هريفنيا أوكراني",
    "displayName-count-many": "هريفنيا أوكراني",
    "displayName-count-other": "هريفنيا أوكراني",
    "symbol": "UAH",
    "symbol-alt-narrow": "₴"
  },
  "UAK": {
    "displayName": "UAK",
    "symbol": "UAK"
  },
  "UGS": {
    "displayName": "شلن أوغندي - 1987-1966",
    "symbol": "UGS"
  },
  "UGX": {
    "displayName": "شلن أوغندي",
    "displayName-count-zero": "شلن أوغندي",
    "displayName-count-one": "شلن أوغندي",
    "displayName-count-two": "شلن أوغندي",
    "displayName-count-few": "شلن أوغندي",
    "displayName-count-many": "شلن أوغندي",
    "displayName-count-other": "شلن أوغندي",
    "symbol": "UGX"
  },
  "USD": {
    "displayName": "دولار أمريكي",
    "displayName-count-zero": "دولار أمريكي",
    "displayName-count-one": "دولار أمريكي",
    "displayName-count-two": "دولار أمريكي",
    "displayName-count-few": "دولار أمريكي",
    "displayName-count-many": "دولار أمريكي",
    "displayName-count-other": "دولار أمريكي",
    "symbol": "US$",
    "symbol-alt-narrow": "US$"
  },
  "USN": {
    "displayName": "دولار أمريكي (اليوم التالي)",
    "symbol": "USN"
  },
  "USS": {
    "displayName": "دولار أمريكي (نفس اليوم)",
    "symbol": "USS"
  },
  "UYI": {
    "displayName": "UYI",
    "symbol": "UYI"
  },

```



```

"UYP": {
  "displayName": "بیزو اوروجوای - 1993-1975",
  "symbol": "UYP"
},
"UYU": {
  "displayName": "بیزو اوروغوای",
  "displayName-count-zero": "بیزو اوروغوای",
  "displayName-count-one": "بیزو اوروغوای",
  "displayName-count-two": "بیزو اوروغوای",
  "displayName-count-few": "بیزو اوروغوای",
  "displayName-count-many": "بیزو اوروغوای",
  "displayName-count-other": "بیزو اوروغوای",
  "symbol": "UYU",
  "symbol-alt-narrow": "UY$"
},
"UZS": {
  "displayName": "سوم اوزبکستانی",
  "displayName-count-zero": "سوم اوزبکستانی",
  "displayName-count-one": "سوم اوزبکستانی",
  "displayName-count-two": "سوم اوزبکستانی",
  "displayName-count-few": "سوم اوزبکستانی",
  "displayName-count-many": "سوم اوزبکستانی",
  "displayName-count-other": "سوم اوزبکستانی",
  "symbol": "UZS"
},
"VEB": {
  "displayName": "بولیفار فنزوئیلی - 2008-1871",
  "symbol": "VEB"
},
"VEF": {
  "displayName": "بولیفار فنزوئیلی",
  "displayName-count-zero": "بولیفار فنزوئیلی",
  "displayName-count-one": "بولیفار فنزوئیلی",
  "displayName-count-two": "بولیفار فنزوئیلی",
  "displayName-count-few": "بولیفار فنزوئیلی",
  "displayName-count-many": "بولیفار فنزوئیلی",
  "displayName-count-other": "بولیفار فنزوئیلی",
  "symbol": "VEF",
  "symbol-alt-narrow": "Bs"
},
"VND": {
  "displayName": "دونج فیتنامی",
  "displayName-count-zero": "دونج فیتنامی",
  "displayName-count-one": "دونج فیتنامی",
  "displayName-count-two": "دونج فیتنامی",
  "displayName-count-few": "دونج فیتنامی",
  "displayName-count-many": "دونج فیتنامی",
  "displayName-count-other": "دونج فیتنامی",
  "symbol": "₫",
  "symbol-alt-narrow": "₫"
},
"VNN": {
  "displayName": "VNN",
  "symbol": "VNN"
},
"VUV": {
  "displayName": "فاتو فانواتو",

```

```

    "displayName-count-zero": "فاتو فانواتو",
    "displayName-count-one": "فاتو فانواتو",
    "displayName-count-two": "فاتو فانواتو",
    "displayName-count-few": "فاتو فانواتو",
    "displayName-count-many": "فاتو فانواتو",
    "displayName-count-other": "فاتو فانواتو",
    "symbol": "VUV"
  },
  "WST": {
    "displayName": "تالا ساموا",
    "displayName-count-zero": "تالا ساموا",
    "displayName-count-one": "تالا ساموا",
    "displayName-count-two": "تالا ساموا",
    "displayName-count-few": "تالا ساموا",
    "displayName-count-many": "تالا ساموا",
    "displayName-count-other": "تالا ساموا",
    "symbol": "WST"
  },
  "XAF": {
    "displayName": "فرنك وسط أفريقي",
    "displayName-count-zero": "فرنك وسط أفريقي",
    "displayName-count-one": "فرنك وسط أفريقي",
    "displayName-count-two": "فرنك وسط أفريقي",
    "displayName-count-few": "فرنك وسط أفريقي",
    "displayName-count-many": "فرنك وسط أفريقي",
    "displayName-count-other": "فرنك وسط أفريقي",
    "symbol": "FCFA"
  },
  "XAG": {
    "displayName": "فضة",
    "symbol": "XAG"
  },
  "XAU": {
    "displayName": "ذهب",
    "symbol": "XAU"
  },
  "XBA": {
    "displayName": "الوحدة الأوروبية المركبة",
    "symbol": "XBA"
  },
  "XBB": {
    "displayName": "الوحدة المالية الأوروبية",
    "symbol": "XBB"
  },
  "XBC": {
    "displayName": "الوحدة الحسابية الأوروبية",
    "symbol": "XBC"
  },
  "XBD": {
    "displayName": "وحدة الحساب الأوروبية (XBD)",
    "symbol": "XBD"
  },
  "XCD": {
    "displayName": "دولار شرق الكاريبي",
    "displayName-count-zero": "دولار شرق الكاريبي",
    "displayName-count-one": "دولار شرق الكاريبي",
    "displayName-count-two": "دولار شرق الكاريبي",

```

```

    "displayName-count-few": "دولار شرق الكاريبي",
    "displayName-count-many": "دولار شرق الكاريبي",
    "displayName-count-other": "دولار شرق الكاريبي",
    "symbol": "EC$",
    "symbol-alt-narrow": "$"
  },
  "XDR": {
    "displayName": "حقوق السحب الخاصة",
    "symbol": "XDR"
  },
  "XEU": {
    "displayName": "وحدة النقد الأوروبية",
    "symbol": "XEU"
  },
  "XFO": {
    "displayName": "فرنك فرنسي ذهبي",
    "symbol": "XFO"
  },
  "XFU": {
    "displayName": "فرنك فرنسي (UIC)",
    "symbol": "XFU"
  },
  "XOF": {
    "displayName": "فرنك غرب أفريقي",
    "displayName-count-zero": "فرنك غرب أفريقي",
    "displayName-count-one": "فرنك غرب أفريقي",
    "displayName-count-two": "فرنك غرب أفريقي",
    "displayName-count-few": "فرنك غرب أفريقي",
    "displayName-count-many": "فرنك غرب أفريقي",
    "displayName-count-other": "فرنك غرب أفريقي",
    "symbol": "CFA"
  },
  "XPD": {
    "displayName": "بالاديوم",
    "symbol": "XPD"
  },
  "XPF": {
    "displayName": "فرنك سي إف بي",
    "displayName-count-zero": "فرنك سي إف بي",
    "displayName-count-one": "فرنك سي إف بي",
    "displayName-count-two": "فرنك سي إف بي",
    "displayName-count-few": "فرنك سي إف بي",
    "displayName-count-many": "فرنك سي إف بي",
    "displayName-count-other": "فرنك سي إف بي",
    "symbol": "CFPF"
  },
  "XPT": {
    "displayName": "البلاتين",
    "symbol": "XPT"
  },
  "XRE": {
    "displayName": "XRE",
    "symbol": "XRE"
  },
  "XSU": {
    "displayName": "XSU",
    "symbol": "XSU"
  }

```

```

    },
    "XTS": {
      "displayName": "كود اختبار العملة",
      "symbol": "XTS"
    },
    "XUA": {
      "displayName": "XUA",
      "symbol": "XUA"
    },
    "XXX": {
      "displayName": "عملة غير معروفة",
      "displayName-count-zero": "(عملة غير معروفة)",
      "displayName-count-one": "(عملة غير معروفة)",
      "displayName-count-two": "(عملة غير معروفة)",
      "displayName-count-few": "(عملة غير معروفة)",
      "displayName-count-many": "(عملة غير معروفة)",
      "displayName-count-other": "(عملة غير معروفة)",
      "symbol": "***"
    },
    "YDD": {
      "displayName": "دينار يمني",
      "symbol": "YDD"
    },
    "YER": {
      "displayName": "ر.ي.",
      "displayName-count-zero": "ر.ي.",
      "displayName-count-one": "ر.ي.",
      "displayName-count-two": "ر.ي.",
      "displayName-count-few": "ر.ي.",
      "displayName-count-many": "ر.ي.",
      "displayName-count-other": "ر.ي."
    },
    "YUD": {
      "displayName": "دينار يوغسلافي",
      "symbol": "YUD"
    },
    "YUM": {
      "displayName": "YUM",
      "symbol": "YUM"
    },
    "YUN": {
      "displayName": "دينار يوغسلافي قابل للتحويل",
      "symbol": "YUN"
    },
    "YUR": {
      "displayName": "YUR",
      "symbol": "YUR"
    },
    "ZAL": {
      "displayName": "راند جنوب أفريقيا-مالي",
      "symbol": "ZAL"
    },
    "ZAR": {
      "displayName": "راند جنوب أفريقيا",
      "displayName-count-zero": "راند جنوب أفريقيا",
      "displayName-count-one": "راند جنوب أفريقيا",

```

```

        "displayName-count-two": "راند جنوب أفريقيا",
        "displayName-count-few": "راند جنوب أفريقيا",
        "displayName-count-many": "راند جنوب أفريقيا",
        "displayName-count-other": "راند جنوب أفريقيا",
        "symbol": "ZAR",
        "symbol-alt-narrow": "R"
    },
    "ZMK": {
        "displayName": "كواشا زامبي - 2012-1968",
        "displayName-count-zero": "كواشا زامبي - 2012-1968",
        "displayName-count-one": "كواشا زامبي - 2012-1968",
        "displayName-count-two": "كواشا زامبي - 2012-1968",
        "displayName-count-few": "كواشا زامبي - 2012-1968",
        "displayName-count-many": "كواشا زامبي - 2012-1968",
        "displayName-count-other": "كواشا زامبي - 2012-1968",
        "symbol": "ZMK"
    },
    "ZMW": {
        "displayName": "كواشا زامبي",
        "displayName-count-zero": "كواشا زامبي",
        "displayName-count-one": "كواشا زامبي",
        "displayName-count-two": "كواشا زامبي",
        "displayName-count-few": "كواشا زامبي",
        "displayName-count-many": "كواشا زامبي",
        "displayName-count-other": "كواشا زامبي",
        "symbol": "ZMW",
        "symbol-alt-narrow": "ZK"
    },
    "ZRN": {
        "displayName": "زائير زائيري جديد",
        "symbol": "ZRN"
    },
    "ZRZ": {
        "displayName": "زائير زائيري",
        "symbol": "ZRZ"
    },
    "ZWD": {
        "displayName": "دولار زمبابوي",
        "symbol": "ZWD"
    },
    "ZWL": {
        "displayName": "دولار زمبابوي 2009",
        "symbol": "ZWL"
    },
    "ZWR": {
        "displayName": "ZWR",
        "symbol": "ZWR"
    }
}
}
}
```

CURRENCIES.JSX

```

{
  "main";
  {
    "ar";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 12879 $",
          "_cldrVersion";
          "30.0.3";
        }
        "language";
        "ar";
      }
      "numbers";
      {
        "currencies";
        {
          "ADP";
          {
            "displayName";
            "بيستا أندوري",
            "symbol";
            "ADP";
          }
          "AED";
          {
            "displayName";
            "درهم إماراتي",
            "displayName-count-zero";
            "درهم إماراتي",
            "displayName-count-one";
            "درهم إماراتي",
            "displayName-count-two";
            "درهم إماراتي",
            "displayName-count-few";
            "درهم إماراتي",
            "displayName-count-many";
            "درهم إماراتي",
            "displayName-count-other";
            "درهم إماراتي",
            "symbol";
            "د.إ.";
          }
          "AFA";
          {
            "displayName";
            "أفغاني - 2002-1927",
            "symbol";
            "AFA";
          }
          "AFN";
          {
            "displayName";

```

```

        "أفغاني",
        "displayName-count-zero";
        "أفغاني أفغانستان",
        "displayName-count-one";
        "أفغاني أفغانستان",
        "displayName-count-two";
        "أفغاني أفغانستان",
        "displayName-count-few";
        "أفغاني أفغانستان",
        "displayName-count-many";
        "أفغاني أفغانستان",
        "displayName-count-other";
        "أفغاني أفغانستان",
        "symbol";
        "AFN";
    }
    "ALL";
    {
        "displayName";
        "ليك ألباني",
        "displayName-count-zero";
        "ليك ألباني",
        "displayName-count-one";
        "ليك ألباني",
        "displayName-count-two";
        "ليك ألباني",
        "displayName-count-few";
        "ليك ألباني",
        "displayName-count-many";
        "ليك ألباني",
        "displayName-count-other";
        "ليك ألباني",
        "symbol";
        "ALL";
    }
    "AMD";
    {
        "displayName";
        "درام أرميني",
        "displayName-count-zero";
        "درام أرميني",
        "displayName-count-one";
        "درام أرميني",
        "displayName-count-two";
        "درام أرميني",
        "displayName-count-few";
        "درام أرميني",
        "displayName-count-many";
        "درام أرميني",
        "displayName-count-other";
        "درام أرميني",
        "symbol";
        "AMD";
    }
    "ANG";
    {
        "displayName";

```

```

        "غيلدر أنتيلي هولندي",
        "displayName-count-zero";
        "غيلدر أنتيلي هولندي",
        "displayName-count-one";
        "غيلدر أنتيلي هولندي",
        "displayName-count-two";
        "غيلدر أنتيلي هولندي",
        "displayName-count-few";
        "غيلدر أنتيلي هولندي",
        "displayName-count-many";
        "غيلدر أنتيلي هولندي",
        "displayName-count-other";
        "غيلدر أنتيلي هولندي",
        "symbol";
        "ANG";
    }
    "AOA";
    {
        "displayName";
        "كوانزا أنجولي",
        "displayName-count-zero";
        "كوانزا أنجولي",
        "displayName-count-one";
        "كوانزا أنجولي",
        "displayName-count-two";
        "كوانزا أنجولي",
        "displayName-count-few";
        "كوانزا أنجولي",
        "displayName-count-many";
        "كوانزا أنجولي",
        "displayName-count-other";
        "كوانزا أنجولي",
        "symbol";
        "AOA",
        "symbol-alt-narrow";
        "Kz";
    }
    "AOK";
    {
        "displayName";
        "كوانزا أنجولي - 1977-1990",
        "symbol";
        "AOK";
    }
    "AON";
    {
        "displayName";
        "كوانزا أنجولي جديدة - 1990-2000",
        "symbol";
        "AON";
    }
    "AOR";
    {
        "displayName";
        "كوانزا أنجولي معدلة - 1995 - 1999",
        "symbol";
        "AOR";
    }

```



```

    }
    "ARA";
    {
        "displayName";
        "استرال أرجنتيني",
        "symbol";
        "ARA";
    }
    "ARL";
    {
        "displayName";
        "ARL",
        "symbol";
        "ARL";
    }
    "ARM";
    {
        "displayName";
        "ARM",
        "symbol";
        "ARM";
    }
    "ARP";
    {
        "displayName";
        "1985-1983 - أرجنتيني - بيزو",
        "symbol";
        "ARP";
    }
    "ARS";
    {
        "displayName";
        "بيزو أرجنتيني",
        "displayName-count-zero";
        "بيزو أرجنتيني",
        "displayName-count-one";
        "بيزو أرجنتيني",
        "displayName-count-two";
        "بيزو أرجنتيني",
        "displayName-count-few";
        "بيزو أرجنتيني",
        "displayName-count-many";
        "بيزو أرجنتيني",
        "displayName-count-other";
        "بيزو أرجنتيني",
        "symbol";
        "ARS",
        "symbol-alt-narrow";
        "AR$";
    }
    "ATS";
    {
        "displayName";
        "شلن نمساوي",
        "symbol";
        "ATS";
    }
}

```

```

"AUD";
{
  "displayName";
  "دولار أسترالي",
  "displayName-count-zero";
  "دولار أسترالي",
  "displayName-count-one";
  "دولار أسترالي",
  "displayName-count-two";
  "دولار أسترالي",
  "displayName-count-few";
  "دولار أسترالي",
  "displayName-count-many";
  "دولار أسترالي",
  "displayName-count-other";
  "دولار أسترالي",
  "symbol";
  "AU$";
  "symbol-alt-narrow";
  "AU$";
}
"AWG";
{
  "displayName";
  "فلورن أروبي",
  "displayName-count-zero";
  "فلورن أروبي",
  "displayName-count-one";
  "فلورن أروبي",
  "displayName-count-two";
  "فلورن أروبي",
  "displayName-count-few";
  "فلورن أروبي",
  "displayName-count-many";
  "فلورن أروبي",
  "displayName-count-other";
  "فلورن أروبي",
  "symbol";
  "AWG";
}
"AZM";
{
  "displayName";
  "مانات أذربيجاني",
  "symbol";
  "AZM";
}
"AZN";
{
  "displayName";
  "مانات أذربيجان",
  "displayName-count-zero";
  "مانت أذربيجاني",
  "displayName-count-one";
  "مانت أذربيجاني",
  "displayName-count-two";
  "مانت أذربيجاني",

```

```

        "displayName-count-few";
        "مانت أذربيجاني",
        "displayName-count-many";
        "مانت أذربيجاني",
        "displayName-count-other";
        "مانت أذربيجاني",
        "symbol";
        "AZN";
    }
    "BAD";
    {
        "displayName";
        "دينار البوسنة والهرسك",
        "symbol";
        "BAD";
    }
    "BAM";
    {
        "displayName";
        "مارك البوسنة والهرسك قابل للتحويل",
        "displayName-count-zero";
        "مارك البوسنة والهرسك قابل للتحويل",
        "displayName-count-one";
        "مارك البوسنة والهرسك قابل للتحويل",
        "displayName-count-two";
        "مارك البوسنة والهرسك قابل للتحويل",
        "displayName-count-few";
        "مارك البوسنة والهرسك قابل للتحويل",
        "displayName-count-many";
        "مارك البوسنة والهرسك قابل للتحويل",
        "displayName-count-other";
        "مارك البوسنة والهرسك قابل للتحويل",
        "symbol";
        "BAM",
        "symbol-alt-narrow";
        "KM";
    }
    "BAN";
    {
        "displayName";
        "BAN",
        "symbol";
        "BAN";
    }
    "BBD";
    {
        "displayName";
        "دولار بربادوسي",
        "displayName-count-zero";
        "دولار بربادوسي",
        "displayName-count-one";
        "دولار بربادوسي",
        "displayName-count-two";
        "دولار بربادوسي",
        "displayName-count-few";
        "دولار بربادوسي",
        "displayName-count-many";
    }

```

```

        "دولار بربادوسي",
        "displayName-count-other";
        "دولار بربادوسي",
        "symbol";
        "BBD",
        "symbol-alt-narrow";
        "BB$";
    }
    "BDT";
    {
        "displayName";
        "তাকা বঙ্গলাদেশি",
        "displayName-count-zero";
        "তাকা বঙ্গলাদেশি",
        "displayName-count-one";
        "তাকা বঙ্গলাদেশি",
        "displayName-count-two";
        "তাকা বঙ্গলাদেশি",
        "displayName-count-few";
        "তাকা বঙ্গলাদেশি",
        "displayName-count-many";
        "তাকা বঙ্গলাদেশি",
        "displayName-count-other";
        "তাকা বঙ্গলাদেশি",
        "symbol";
        "BDT",
        "symbol-alt-narrow";
        "৳";
    }
    "BEC";
    {
        "displayName";
        "فرنك بلجيكي قابل للتحويل",
        "symbol";
        "BEC";
    }
    "BEF";
    {
        "displayName";
        "فرنك بلجيكي",
        "symbol";
        "BEF";
    }
    "BEL";
    {
        "displayName";
        "فرنك بلجيكي مالي",
        "symbol";
        "BEL";
    }
    "BGL";
    {
        "displayName";
        "BGL",
        "symbol";
        "BGL";
    }

```

```

    }
    "BGM";
    {
        "displayName";
        "BGM",
        "symbol";
        "BGM";
    }
    "BGN";
    {
        "displayName";
        "ليف بلغاري",
        "displayName-count-zero";
        "ليف بلغاري",
        "displayName-count-one";
        "ليف بلغاري",
        "displayName-count-two";
        "ليف بلغاري",
        "displayName-count-few";
        "ليف بلغاري",
        "displayName-count-many";
        "ليف بلغاري",
        "displayName-count-other";
        "ليف بلغاري",
        "symbol";
        "BGN";
    }
    "BGO";
    {
        "displayName";
        "BGO",
        "symbol";
        "BGO";
    }
    "BHD";
    {
        "displayName";
        "دينار بحريني",
        "displayName-count-zero";
        "دينار بحريني",
        "displayName-count-one";
        "دينار بحريني",
        "displayName-count-two";
        "دينار بحريني",
        "displayName-count-few";
        "دينار بحريني",
        "displayName-count-many";
        "دينار بحريني",
        "displayName-count-other";
        "دينار بحريني",
        "symbol";
        "د.ب.";
    }
    "BIF";
    {
        "displayName";
        "فرنك بروندي",

```

```

        "displayName-count-zero";
        "فرنك بروندي",
        "displayName-count-one";
        "فرنك بروندي",
        "displayName-count-two";
        "فرنك بروندي",
        "displayName-count-few";
        "فرنك بروندي",
        "displayName-count-many";
        "فرنك بروندي",
        "displayName-count-other";
        "فرنك بروندي",
        "symbol";
        "BIF";
    }
    "BMD";
    {
        "displayName";
        "دولار برمودي",
        "displayName-count-zero";
        "دولار برمودي",
        "displayName-count-one";
        "دولار برمودي",
        "displayName-count-two";
        "دولار برمودي",
        "displayName-count-few";
        "دولار برمودي",
        "displayName-count-many";
        "دولار برمودي",
        "displayName-count-other";
        "دولار برمودي",
        "symbol";
        "BMD",
        "symbol-alt-narrow";
        "BM$";
    }
    "BND";
    {
        "displayName";
        "دولار برونائي",
        "displayName-count-zero";
        "دولار برونائي",
        "displayName-count-one";
        "دولار برونائي",
        "displayName-count-two";
        "دولار برونائي",
        "displayName-count-few";
        "دولار برونائي",
        "displayName-count-many";
        "دولار برونائي",
        "displayName-count-other";
        "دولار برونائي",
        "symbol";
        "BND",
        "symbol-alt-narrow";
        "BN$";
    }
}

```

```

"BOB";
{
  "displayName";
  "بولىفيانو بولىفي",
  "displayName-count-zero";
  "بولىفيانو بولىفي",
  "displayName-count-one";
  "بولىفيانو بولىفي",
  "displayName-count-two";
  "بولىفيانو بولىفي",
  "displayName-count-few";
  "بولىفيانو بولىفي",
  "displayName-count-many";
  "بولىفيانو بولىفي",
  "displayName-count-other";
  "بولىفيانو بولىفي",
  "symbol";
  "BOB",
  "symbol-alt-narrow";
  "Bs";
}
"BOL";
{
  "displayName";
  "BOL",
  "symbol";
  "BOL";
}
"BOP";
{
  "displayName";
  "بىزو بولىفي",
  "symbol";
  "BOP";
}
"BOV";
{
  "displayName";
  "مفدول بولىفي",
  "symbol";
  "BOV";
}
"BRB";
{
  "displayName";
  "نوفو كروزايرو برازيلى - 1986-1967",
  "symbol";
  "BRB";
}
"BRC";
{
  "displayName";
  "كروزادو برازيلى",
  "symbol";
  "BRC";
}
"BRE";

```

```

    {
      "displayName";
      "1993-1990 - كروزايرو برازيلى",
      "symbol";
      "BRE";
    }
    "BRL";
    {
      "displayName";
      "ڤل برازيلى",
      "displayName-count-zero";
      "ڤل برازيلى",
      "displayName-count-one";
      "ڤل برازيلى",
      "displayName-count-two";
      "ڤل برازيلى",
      "displayName-count-few";
      "ڤل برازيلى",
      "displayName-count-many";
      "ڤل برازيلى",
      "displayName-count-other";
      "ڤل برازيلى",
      "symbol";
      "R$",
      "symbol-alt-narrow";
      "R$";
    }
    "BRN";
    {
      "displayName";
      "BRN",
      "symbol";
      "BRN";
    }
    "BRR";
    {
      "displayName";
      "BRR",
      "symbol";
      "BRR";
    }
    "BRZ";
    {
      "displayName";
      "BRZ",
      "symbol";
      "BRZ";
    }
    "BSD";
    {
      "displayName";
      "دولار باهامى",
      "displayName-count-zero";
      "دولار باهامى",
      "displayName-count-one";
      "دولار باهامى",
      "displayName-count-two";
    }

```



```

        "دولار باهامي",
        "displayName-count-few";
        "دولار باهامي",
        "displayName-count-many";
        "دولار باهامي",
        "displayName-count-other";
        "دولار باهامي",
        "symbol";
        "BSD",
        "symbol-alt-narrow";
        "BS$";
    }
    "BTN";
    {
        "displayName";
        "نولتوم بوتاني",
        "displayName-count-zero";
        "نولتوم بوتاني",
        "displayName-count-one";
        "نولتوم بوتاني",
        "displayName-count-two";
        "نولتوم بوتاني",
        "displayName-count-few";
        "نولتوم بوتاني",
        "displayName-count-many";
        "نولتوم بوتاني",
        "displayName-count-other";
        "نولتوم بوتاني",
        "symbol";
        "BTN";
    }
    "BUK";
    {
        "displayName";
        "کيات بورمي",
        "symbol";
        "BUK";
    }
    "BWP";
    {
        "displayName";
        "بولا بتسواني",
        "displayName-count-zero";
        "بولا بتسواني",
        "displayName-count-one";
        "بولا بتسواني",
        "displayName-count-two";
        "بولا بتسواني",
        "displayName-count-few";
        "بولا بتسواني",
        "displayName-count-many";
        "بولا بتسواني",
        "displayName-count-other";
        "بولا بتسواني",
        "symbol";
        "BWP",
        "symbol-alt-narrow";
    }

```

```

        "P";
    }
    "BYB";
    {
        "displayName";
        "1999-1994 - روبل بيلاروسي جديد",
        "symbol";
        "BYB";
    }
    "BYN";
    {
        "displayName";
        "روبل بيلاروسي",
        "displayName-count-zero";
        "روبل بيلاروسي",
        "displayName-count-one";
        "روبل بيلاروسي",
        "displayName-count-two";
        "روبل بيلاروسي",
        "displayName-count-few";
        "روبل بيلاروسي",
        "displayName-count-many";
        "روبل بيلاروسي",
        "displayName-count-other";
        "روبل بيلاروسي",
        "symbol";
        "BYN",
        "symbol-alt-narrow";
        "p.";
    }
    "BYR";
    {
        "displayName";
        "2016-2000 (روبل بيلاروسي)",
        "displayName-count-zero";
        "2016-2000 (روبل بيلاروسي)",
        "displayName-count-one";
        "2016-2000 (روبل بيلاروسي)",
        "displayName-count-two";
        "2016-2000 (روبل بيلاروسي)",
        "displayName-count-few";
        "2016-2000 (روبل بيلاروسي)",
        "displayName-count-many";
        "2016-2000 (روبل بيلاروسي)",
        "displayName-count-other";
        "2016-2000 (روبل بيلاروسي)",
        "symbol";
        "BYR";
    }
    "BZD";
    {
        "displayName";
        "دولار بليزي",
        "displayName-count-zero";
        "دولار بليزي",
        "displayName-count-one";
        "دولار بليزي",
    }

```

```

        "displayName-count-two";
        "دولاران بليزيان",
        "displayName-count-few";
        "دولار بليزي",
        "displayName-count-many";
        "دولار بليزي",
        "displayName-count-other";
        "دولار بليزي",
        "symbol";
        "BZD",
        "symbol-alt-narrow";
        "BZ$";
    }
    "CAD";
    {
        "displayName";
        "دولار كندي",
        "displayName-count-zero";
        "دولار كندي",
        "displayName-count-one";
        "دولار كندي",
        "displayName-count-two";
        "دولار كندي",
        "displayName-count-few";
        "دولار كندي",
        "displayName-count-many";
        "دولار كندي",
        "displayName-count-other";
        "دولار كندي",
        "symbol";
        "CA$",
        "symbol-alt-narrow";
        "CA$";
    }
    "CDF";
    {
        "displayName";
        "فرنك كونغولي",
        "displayName-count-zero";
        "فرنك كونغولي",
        "displayName-count-one";
        "فرنك كونغولي",
        "displayName-count-two";
        "فرنك كونغولي",
        "displayName-count-few";
        "فرنك كونغولي",
        "displayName-count-many";
        "فرنك كونغولي",
        "displayName-count-other";
        "فرنك كونغولي",
        "symbol";
        "CDF";
    }
    "CHE";
    {
        "displayName";
        "CHE",

```

```

        "symbol";
        "CHE";
    }
    "CHF";
    {
        "displayName";
        "فرنك سويسري",
        "displayName-count-zero";
        "فرنك سويسري",
        "displayName-count-one";
        "فرنك سويسري",
        "displayName-count-two";
        "فرنك سويسري",
        "displayName-count-few";
        "فرنك سويسري",
        "displayName-count-many";
        "فرنك سويسري",
        "displayName-count-other";
        "فرنك سويسري",
        "symbol";
        "CHF";
    }
    "CHW";
    {
        "displayName";
        "CHW",
        "symbol";
        "CHW";
    }
    "CLE";
    {
        "displayName";
        "CLE",
        "symbol";
        "CLE";
    }
    "CLF";
    {
        "displayName";
        "CLF",
        "symbol";
        "CLF";
    }
    "CLP";
    {
        "displayName";
        "بيزو شيلي",
        "displayName-count-zero";
        "بيزو شيلي",
        "displayName-count-one";
        "بيزو شيلي",
        "displayName-count-two";
        "بيزو شيلي",
        "displayName-count-few";
        "بيزو شيلي",
        "displayName-count-many";
        "بيزو شيلي",
    }

```

```

        "displayName-count-other";
        "بيزو شيلي",
        "symbol";
        "CLP",
        "symbol-alt-narrow";
        "CL$";
    }
    "CNY";
    {
        "displayName";
        "يوان صيني",
        "displayName-count-zero";
        "يوان صيني",
        "displayName-count-one";
        "يوان صيني",
        "displayName-count-two";
        "يوان صيني",
        "displayName-count-few";
        "يوان صيني",
        "displayName-count-many";
        "يوان صيني",
        "displayName-count-other";
        "يوان صيني",
        "symbol";
        "CN¥",
        "symbol-alt-narrow";
        "CN¥";
    }
    "COP";
    {
        "displayName";
        "بيزو كولومبي",
        "displayName-count-zero";
        "بيزو كولومبي",
        "displayName-count-one";
        "بيزو كولومبي",
        "displayName-count-two";
        "بيزو كولومبي",
        "displayName-count-few";
        "بيزو كولومبي",
        "displayName-count-many";
        "بيزو كولومبي",
        "displayName-count-other";
        "بيزو كولومبي",
        "symbol";
        "COP",
        "symbol-alt-narrow";
        "CO$";
    }
    "COU";
    {
        "displayName";
        "COU",
        "symbol";
        "COU";
    }
    "CRC";

```

```

{
  "displayName";
  "کولن کوستا ريکي",
  "displayName-count-zero";
  "کولن کوستا ريکي",
  "displayName-count-one";
  "کولن کوستا ريکي",
  "displayName-count-two";
  "کولن کوستا ريکي",
  "displayName-count-few";
  "کولن کوستا ريکي",
  "displayName-count-many";
  "کولن کوستا ريکي",
  "displayName-count-other";
  "کولن کوستا ريکي",
  "symbol";
  "CRC",
  "symbol-alt-narrow";
  "¢";
}
"CSD";
{
  "displayName";
  "دينار صربي قديم",
  "symbol";
  "CSD";
}
"CSK";
{
  "displayName";
  "کرونة تشيکوسلوفاکيا",
  "symbol";
  "CSK";
}
"CUC";
{
  "displayName";
  "بیزو کوبي قابل للتحويل",
  "displayName-count-zero";
  "بیزو کوبي قابل للتحويل",
  "displayName-count-one";
  "بیزو کوبي قابل للتحويل",
  "displayName-count-two";
  "بیزو کوبي قابل للتحويل",
  "displayName-count-few";
  "بیزو کوبي قابل للتحويل",
  "displayName-count-many";
  "بیزو کوبي قابل للتحويل",
  "displayName-count-other";
  "بیزو کوبي قابل للتحويل",
  "symbol";
  "CUC",
  "symbol-alt-narrow";
  "$";
}
"CUP";
{

```

```

        "displayName";
        "بيزو كوبي",
        "displayName-count-zero";
        "بيزو كوبي",
        "displayName-count-one";
        "بيزو كوبي",
        "displayName-count-two";
        "بيزو كوبي",
        "displayName-count-few";
        "بيزو كوبي",
        "displayName-count-many";
        "بيزو كوبي",
        "displayName-count-other";
        "بيزو كوبي",
        "symbol";
        "CUP",
        "symbol-alt-narrow";
        "CU$";
    }
    "CVE";
    {
        "displayName";
        "اسكودو الرأس الخضراء",
        "displayName-count-zero";
        "اسكودو الرأس الخضراء",
        "displayName-count-one";
        "اسكودو الرأس الخضراء",
        "displayName-count-two";
        "اسكودو الرأس الخضراء",
        "displayName-count-few";
        "اسكودو الرأس الخضراء",
        "displayName-count-many";
        "اسكودو الرأس الخضراء",
        "displayName-count-other";
        "اسكودو الرأس الخضراء",
        "symbol";
        "CVE";
    }
    "CYP";
    {
        "displayName";
        "جنيه قبرصي",
        "symbol";
        "CYP";
    }
    "CZK";
    {
        "displayName";
        "كرونة تشيكية",
        "displayName-count-zero";
        "كرونة تشيكية",
        "displayName-count-one";
        "كرونة تشيكية",
        "displayName-count-two";
        "كرونة تشيكية",
        "displayName-count-few";
        "كرونة تشيكية",
    }

```

```

        "displayName-count-many";
        "كرونة تشيكية",
        "displayName-count-other";
        "كرونة تشيكية",
        "symbol";
        "CZK",
        "symbol-alt-narrow";
        "Kč";
    }
    "DDM";
    {
        "displayName";
        "أوستمارك ألماني شرقي",
        "symbol";
        "DDM";
    }
    "DEM";
    {
        "displayName";
        "مارك ألماني",
        "symbol";
        "DEM";
    }
    "DJF";
    {
        "displayName";
        "فرنك جيبوتي",
        "displayName-count-zero";
        "فرنك جيبوتي",
        "displayName-count-one";
        "فرنك جيبوتي",
        "displayName-count-two";
        "فرنك جيبوتي",
        "displayName-count-few";
        "فرنك جيبوتي",
        "displayName-count-many";
        "فرنك جيبوتي",
        "displayName-count-other";
        "فرنك جيبوتي",
        "symbol";
        "DJF";
    }
    "DKK";
    {
        "displayName";
        "كرونة دانماركي",
        "displayName-count-zero";
        "كرونة دانماركي",
        "displayName-count-one";
        "كرونة دانماركي",
        "displayName-count-two";
        "كرونة دانماركي",
        "displayName-count-few";
        "كرونة دانماركي",
        "displayName-count-many";
        "كرونة دانماركي",
        "displayName-count-other";
    }

```



```

        "كرونة دانماركي",
        "symbol";
        "DKK",
        "symbol-alt-narrow";
        "kr";
    }
    "DOP";
    {
        "displayName";
        "بيزو الدومنيكان",
        "displayName-count-zero";
        "بيزو الدومنيكان",
        "displayName-count-one";
        "بيزو الدومنيكان",
        "displayName-count-two";
        "بيزو الدومنيكان",
        "displayName-count-few";
        "بيزو الدومنيكان",
        "displayName-count-many";
        "بيزو الدومنيكان",
        "displayName-count-other";
        "بيزو الدومنيكان",
        "symbol";
        "DOP",
        "symbol-alt-narrow";
        "DO$";
    }
    "DZD";
    {
        "displayName";
        "دينار جزائري",
        "displayName-count-zero";
        "دينار جزائري",
        "displayName-count-one";
        "دينار جزائري",
        "displayName-count-two";
        "ديناران جزائريان",
        "displayName-count-few";
        "دينارات جزائرية",
        "displayName-count-many";
        "دينارًا جزائريًا",
        "displayName-count-other";
        "دينار جزائري",
        "symbol";
        "د.ج.";
    }
    "ECS";
    {
        "displayName";
        "ECS",
        "symbol";
        "ECS";
    }
    "ECV";
    {
        "displayName";
        "ECV",

```

```

        "symbol";
        "ECV";
    }
    "EEK";
    {
        "displayName";
        "كرونه استونية",
        "symbol";
        "EEK";
    }
    "EGP";
    {
        "displayName";
        "جنيه مصري",
        "displayName-count-zero";
        "جنيه مصري",
        "displayName-count-one";
        "جنيه مصري",
        "displayName-count-two";
        "جنيهان مصريان",
        "displayName-count-few";
        "جنيهات مصرية",
        "displayName-count-many";
        "جنيهاً مصرياً",
        "displayName-count-other";
        "جنيه مصري",
        "symbol";
        "ج.م.ع",
        "symbol-alt-narrow";
        "E£";
    }
    "ERN";
    {
        "displayName";
        "ناكفا أريتري",
        "displayName-count-zero";
        "ناكفا أريتري",
        "displayName-count-one";
        "ناكفا أريتري",
        "displayName-count-two";
        "ناكفا أريتري",
        "displayName-count-few";
        "ناكفا أريتري",
        "displayName-count-many";
        "ناكفا أريتري",
        "displayName-count-other";
        "ناكفا أريتري",
        "symbol";
        "ERN";
    }
    "ESA";
    {
        "displayName";
        "ESA",
        "symbol";
        "ESA";
    }
}

```

```

"ESB";
{
  "displayName";
  "ESB",
  "symbol";
  "ESB";
}
"ESP";
{
  "displayName";
  "بیزیتا إسبانی",
  "symbol";
  "ESP",
  "symbol-alt-narrow";
  "₧";
}
"ETB";
{
  "displayName";
  "بیر أثیوبی",
  "displayName-count-zero";
  "بیر أثیوبی",
  "displayName-count-one";
  "بیر أثیوبی",
  "displayName-count-two";
  "بیر أثیوبی",
  "displayName-count-few";
  "بیر أثیوبی",
  "displayName-count-many";
  "بیر أثیوبی",
  "displayName-count-other";
  "بیر أثیوبی",
  "symbol";
  "ETB";
}
"EUR";
{
  "displayName";
  "یورو",
  "displayName-count-zero";
  "یورو",
  "displayName-count-one";
  "یورو",
  "displayName-count-two";
  "یورو",
  "displayName-count-few";
  "یورو",
  "displayName-count-many";
  "یورو",
  "displayName-count-other";
  "یورو",
  "symbol";
  "€",
  "symbol-alt-narrow";
  "€";
}
"FIM";

```

```

    {
      "displayName";
      "ماركا فنلندي",
      "symbol";
      "FIM";
    }
    "FJD";
    {
      "displayName";
      "دولار فيجي",
      "displayName-count-zero";
      "دولار فيجي",
      "displayName-count-one";
      "دولار فيجي",
      "displayName-count-two";
      "دولار فيجي",
      "displayName-count-few";
      "دولار فيجي",
      "displayName-count-many";
      "دولار فيجي",
      "displayName-count-other";
      "دولار فيجي",
      "symbol";
      "FJD",
      "symbol-alt-narrow";
      "FJ$";
    }
    "FKP";
    {
      "displayName";
      "جنيه جزر فوكلاند",
      "displayName-count-zero";
      "جنيه جزر فوكلاند",
      "displayName-count-one";
      "جنيه جزر فوكلاند",
      "displayName-count-two";
      "جنيه جزر فوكلاند",
      "displayName-count-few";
      "جنيه جزر فوكلاند",
      "displayName-count-many";
      "جنيه جزر فوكلاند",
      "displayName-count-other";
      "جنيه جزر فوكلاند",
      "symbol";
      "FKP",
      "symbol-alt-narrow";
      "£";
    }
    "FRF";
    {
      "displayName";
      "فرنك فرنسي",
      "symbol";
      "FRF";
    }
    "GBP";
    {

```

```

        "displayName";
        "جنيه إسترليني",
        "displayName-count-zero";
        "جنيه إسترليني",
        "displayName-count-one";
        "جنيه إسترليني",
        "displayName-count-two";
        "جنيه إسترليني",
        "displayName-count-few";
        "جنيه إسترليني",
        "displayName-count-many";
        "جنيه إسترليني",
        "displayName-count-other";
        "جنيه إسترليني",
        "symbol";
        "£",
        "symbol-alt-narrow";
        "UK£";
    }
    "GEK";
    {
        "displayName";
        "GEK",
        "symbol";
        "GEK";
    }
    "GEL";
    {
        "displayName";
        "ლარი جورجი",
        "displayName-count-zero";
        "ლარი جورجი",
        "displayName-count-one";
        "ლარი جورجი",
        "displayName-count-two";
        "ლარი جورجი",
        "displayName-count-few";
        "ლარი جورجი",
        "displayName-count-many";
        "ლარი جورجი",
        "displayName-count-other";
        "ლარი جورجი",
        "symbol";
        "GEL",
        "symbol-alt-narrow";
        "ლ",
        "symbol-alt-variant";
        "ლ";
    }
    "GHC";
    {
        "displayName";
        "სიდი განი",
        "symbol";
        "GHC";
    }
    "GHS";

```

```

{
  "displayName";
  "سيدي غانا",
  "displayName-count-zero";
  "سيدي غانا",
  "displayName-count-one";
  "سيدي غانا",
  "displayName-count-two";
  "سيدي غانا",
  "displayName-count-few";
  "سيدي غانا",
  "displayName-count-many";
  "سيدي غانا",
  "displayName-count-other";
  "سيدي غانا",
  "symbol";
  "GHS";
}
"GIP";
{
  "displayName";
  "جنيه جبل طارق",
  "displayName-count-zero";
  "جنيه جبل طارق",
  "displayName-count-one";
  "جنيه جبل طارق",
  "displayName-count-two";
  "جنيه جبل طارق",
  "displayName-count-few";
  "جنيه جبل طارق",
  "displayName-count-many";
  "جنيه جبل طارق",
  "displayName-count-other";
  "جنيه جبل طارق",
  "symbol";
  "GIP",
  "symbol-alt-narrow";
  "£";
}
"GMD";
{
  "displayName";
  "دلّاسي جامبي",
  "displayName-count-zero";
  "دلّاسي جامبي",
  "displayName-count-one";
  "دلّاسي جامبي",
  "displayName-count-two";
  "دلّاسي جامبي",
  "displayName-count-few";
  "دلّاسي جامبي",
  "displayName-count-many";
  "دلّاسي جامبي",
  "displayName-count-other";
  "دلّاسي جامبي",
  "symbol";
  "GMD";
}

```

```

    }
    "GNF";
    {
        "displayName";
        "فرنك غينيا",
        "displayName-count-zero";
        "فرنك غينيا",
        "displayName-count-one";
        "فرنك غينيا",
        "displayName-count-two";
        "فرنك غينيا",
        "displayName-count-few";
        "فرنك غينيا",
        "displayName-count-many";
        "فرنك غينيا",
        "displayName-count-other";
        "فرنك غينيا",
        "symbol";
        "GNF",
        "symbol-alt-narrow";
        "FG";
    }
    "GNS";
    {
        "displayName";
        "سيلي غينيا",
        "symbol";
        "GNS";
    }
    "GQE";
    {
        "displayName";
        "اكويل جونيونا غينيا الاستوائية",
        "symbol";
        "GQE";
    }
    "GRD";
    {
        "displayName";
        "دراخما يوناني",
        "symbol";
        "GRD";
    }
    "GTQ";
    {
        "displayName";
        "كوتزال جواتيمالا",
        "displayName-count-zero";
        "كوتزال جواتيمالا",
        "displayName-count-one";
        "كوتزال جواتيمالا",
        "displayName-count-two";
        "كوتزال جواتيمالا",
        "displayName-count-few";
        "كوتزال جواتيمالا",
        "displayName-count-many";
        "كوتزال جواتيمالا",
    }

```

```

        "displayName-count-other";
        "كوتزال جواتيمالا",
        "symbol";
        "GTQ",
        "symbol-alt-narrow";
        "Q";
    }
    "GWE";
    {
        "displayName";
        "اسكود برتغالي غينيا",
        "symbol";
        "GWE";
    }
    "GWP";
    {
        "displayName";
        "بيزو غينيا بيساو",
        "symbol";
        "GWP";
    }
    "GYD";
    {
        "displayName";
        "دولار غيانا",
        "displayName-count-zero";
        "دولار غيانا",
        "displayName-count-one";
        "دولار غيانا",
        "displayName-count-two";
        "دولار غيانا",
        "displayName-count-few";
        "دولار غيانا",
        "displayName-count-many";
        "دولار غيانا",
        "displayName-count-other";
        "دولار غيانا",
        "symbol";
        "GYD",
        "symbol-alt-narrow";
        "GY$";
    }
    "HKD";
    {
        "displayName";
        "دولار هونغ كونغ",
        "displayName-count-zero";
        "دولار هونغ كونغ",
        "displayName-count-one";
        "دولار هونغ كونغ",
        "displayName-count-two";
        "دولار هونغ كونغ",
        "displayName-count-few";
        "دولار هونغ كونغ",
        "displayName-count-many";
        "دولار هونغ كونغ",
        "displayName-count-other";
    }

```



```

        "دولار هونغ كونغ",
        "symbol";
        "HK$",
        "symbol-alt-narrow";
        "HK$";
    }
    "HNL";
    {
        "displayName";
        "ليمبيرا هندوراس",
        "displayName-count-zero";
        "ليمبيرا هندوراس",
        "displayName-count-one";
        "ليمبيرا هندوراس",
        "displayName-count-two";
        "ليمبيرا هندوراس",
        "displayName-count-few";
        "ليمبيرا هندوراس",
        "displayName-count-many";
        "ليمبيرا هندوراس",
        "displayName-count-other";
        "ليمبيرا هندوراس",
        "symbol";
        "HNL",
        "symbol-alt-narrow";
        "L";
    }
    "HRD";
    {
        "displayName";
        "دينار كرواتي",
        "symbol";
        "HRD";
    }
    "HRK";
    {
        "displayName";
        "كونا كرواتي",
        "displayName-count-zero";
        "كونا كرواتي",
        "displayName-count-one";
        "كونا كرواتي",
        "displayName-count-two";
        "كونا كرواتي",
        "displayName-count-few";
        "كونا كرواتي",
        "displayName-count-many";
        "كونا كرواتي",
        "displayName-count-other";
        "كونا كرواتي",
        "symbol";
        "HRK",
        "symbol-alt-narrow";
        "kn";
    }
    "HTG";
    {

```

```

        "displayName";
        "جوردی هایتي",
        "displayName-count-zero";
        "جوردی هایتي",
        "displayName-count-one";
        "جوردی هایتي",
        "displayName-count-two";
        "جوردی هایتي",
        "displayName-count-few";
        "جوردی هایتي",
        "displayName-count-many";
        "جوردی هایتي",
        "displayName-count-other";
        "جوردی هایتي",
        "symbol";
        "HTG";
    }
    "HUF";
    {
        "displayName";
        "فورينت مجري",
        "displayName-count-zero";
        "فورينت مجري",
        "displayName-count-one";
        "فورينت مجري",
        "displayName-count-two";
        "فورينت مجري",
        "displayName-count-few";
        "فورينت مجري",
        "displayName-count-many";
        "فورينت مجري",
        "displayName-count-other";
        "فورينت مجري",
        "symbol";
        "HUF",
        "symbol-alt-narrow";
        "Ft";
    }
    "IDR";
    {
        "displayName";
        "روبية إندونيسية",
        "displayName-count-zero";
        "روبية إندونيسية",
        "displayName-count-one";
        "روبية إندونيسية",
        "displayName-count-two";
        "روبية إندونيسية",
        "displayName-count-few";
        "روبية إندونيسية",
        "displayName-count-many";
        "روبية إندونيسية",
        "displayName-count-other";
        "روبية إندونيسية",
        "symbol";
        "IDR",
        "symbol-alt-narrow";
    }

```

```

        "Rp";
    }
    "IEP";
    {
        "displayName";
        "جنيه إيرلندي",
        "symbol";
        "IEP";
    }
    "ILP";
    {
        "displayName";
        "جنيه إسرائيلي",
        "symbol";
        "ILP";
    }
    "ILS";
    {
        "displayName";
        "شيكل إسرائيلي جديد",
        "displayName-count-zero";
        "شيكل إسرائيلي جديد",
        "displayName-count-one";
        "شيكل إسرائيلي جديد",
        "displayName-count-two";
        "شيكل إسرائيلي جديد",
        "displayName-count-few";
        "شيكل إسرائيلي جديد",
        "displayName-count-many";
        "شيكل إسرائيلي جديد",
        "displayName-count-other";
        "شيكل إسرائيلي جديد",
        "symbol";
        "₪",
        "symbol-alt-narrow";
        "₪";
    }
    "INR";
    {
        "displayName";
        "روبية هندي",
        "displayName-count-zero";
        "روبية هندي",
        "displayName-count-one";
        "روبية هندي",
        "displayName-count-two";
        "روبية هندي",
        "displayName-count-few";
        "روبية هندي",
        "displayName-count-many";
        "روبية هندي",
        "displayName-count-other";
        "روبية هندي",
        "symbol";
        "₹",
        "symbol-alt-narrow";
        "₹";
    }

```

```

    }
    "IQD";
    {
        "displayName";
        "دينار عراقي",
        "displayName-count-zero";
        "دينار عراقي",
        "displayName-count-one";
        "دينار عراقي",
        "displayName-count-two";
        "دينار عراقي",
        "displayName-count-few";
        "دينار عراقي",
        "displayName-count-many";
        "دينار عراقي",
        "displayName-count-other";
        "دينار عراقي",
        "symbol";
        "د.ع.";
    }
    "IRR";
    {
        "displayName";
        "ریال ایرانی",
        "displayName-count-zero";
        "ریال ایرانی",
        "displayName-count-one";
        "ریال ایرانی",
        "displayName-count-two";
        "ریال ایرانی",
        "displayName-count-few";
        "ریال ایرانی",
        "displayName-count-many";
        "ریال ایرانی",
        "displayName-count-other";
        "ریال ایرانی",
        "symbol";
        "ر.ا.";
    }
    "ISK";
    {
        "displayName";
        "کرونة ايسلندي",
        "displayName-count-zero";
        "کرونة ايسلندي",
        "displayName-count-one";
        "کرونة ايسلندي",
        "displayName-count-two";
        "کرونة ايسلندي",
        "displayName-count-few";
        "کرونة ايسلندي",
        "displayName-count-many";
        "کرونة ايسلندي",
        "displayName-count-other";
        "کرونة ايسلندي",
        "symbol";
        "ISK",
    }

```

```

        "symbol-alt-narrow";
        "kr";
    }
    "ITL";
    {
        "displayName";
        "ليرة إيطالية",
        "symbol";
        "ITL";
    }
    "JMD";
    {
        "displayName";
        "دولار جامايكي",
        "displayName-count-zero";
        "دولار جامايكي",
        "displayName-count-one";
        "دولار جامايكي",
        "displayName-count-two";
        "دولار جامايكي",
        "displayName-count-few";
        "دولار جامايكي",
        "displayName-count-many";
        "دولار جامايكي",
        "displayName-count-other";
        "دولار جامايكي",
        "symbol";
        "JMD",
        "symbol-alt-narrow";
        "JM$";
    }
    "JOD";
    {
        "displayName";
        "دينار أردني",
        "displayName-count-zero";
        "دينار أردني",
        "displayName-count-one";
        "دينار أردني",
        "displayName-count-two";
        "دينار أردني",
        "displayName-count-few";
        "دينار أردني",
        "displayName-count-many";
        "دينار أردني",
        "displayName-count-other";
        "دينار أردني",
        "symbol";
        "د.أ.";
    }
    "JPY";
    {
        "displayName";
        "ين ياباني",
        "displayName-count-zero";
        "ين ياباني",
        "displayName-count-one";
    }

```

```

        "ين ياباني",
        "displayName-count-two";
        "ين ياباني",
        "displayName-count-few";
        "ين ياباني",
        "displayName-count-many";
        "ين ياباني",
        "displayName-count-other";
        "ين ياباني",
        "symbol";
        "JP¥",
        "symbol-alt-narrow";
        "JP¥";
    }
    "KES";
    {
        "displayName";
        "شلن كينيي",
        "displayName-count-zero";
        "شلن كينيي",
        "displayName-count-one";
        "شلن كينيي",
        "displayName-count-two";
        "شلن كينيي",
        "displayName-count-few";
        "شلن كينيي",
        "displayName-count-many";
        "شلن كينيي",
        "displayName-count-other";
        "شلن كينيي",
        "symbol";
        "KES";
    }
    "KGS";
    {
        "displayName";
        "سوم قيرغستاني",
        "displayName-count-zero";
        "سوم قيرغستاني",
        "displayName-count-one";
        "سوم قيرغستاني",
        "displayName-count-two";
        "سوم قيرغستاني",
        "displayName-count-few";
        "سوم قيرغستاني",
        "displayName-count-many";
        "سوم قيرغستاني",
        "displayName-count-other";
        "سوم قيرغستاني",
        "symbol";
        "KGS";
    }
    "KHR";
    {
        "displayName";
        "ريال كمبودي",
        "displayName-count-zero";

```

```

        "ريال كمبودي",
        "displayName-count-one";
        "ريال كمبودي",
        "displayName-count-two";
        "ريال كمبودي",
        "displayName-count-few";
        "ريال كمبودي",
        "displayName-count-many";
        "ريال كمبودي",
        "displayName-count-other";
        "ريال كمبودي",
        "symbol";
        "KHR",
        "symbol-alt-narrow";
        "฿";
    }
    "KMF";
    {
        "displayName";
        "فرنك جزر القمر",
        "displayName-count-zero";
        "فرنك جزر القمر",
        "displayName-count-one";
        "فرنك جزر القمر",
        "displayName-count-two";
        "فرنك جزر القمر",
        "displayName-count-few";
        "فرنك جزر القمر",
        "displayName-count-many";
        "فرنك جزر القمر",
        "displayName-count-other";
        "فرنك جزر القمر",
        "symbol";
        "ف.ج.ق.",
        "symbol-alt-narrow";
        "CF";
    }
    "KPW";
    {
        "displayName";
        "وون كوريا الشمالية",
        "displayName-count-zero";
        "وون كوريا الشمالية",
        "displayName-count-one";
        "وون كوريا الشمالية",
        "displayName-count-two";
        "وون كوريا الشمالية",
        "displayName-count-few";
        "وون كوريا الشمالية",
        "displayName-count-many";
        "وون كوريا الشمالية",
        "displayName-count-other";
        "وون كوريا الشمالية",
        "symbol";
        "KPW",
        "symbol-alt-narrow";
    }

```

```

        "₩";
    }
    "KRH";
    {
        "displayName";
        "KRH",
        "symbol";
        "KRH";
    }
    "KRO";
    {
        "displayName";
        "KRO",
        "symbol";
        "KRO";
    }
    "KRW";
    {
        "displayName";
        "₩",
        "displayName-count-zero";
        "₩",
        "displayName-count-one";
        "₩",
        "displayName-count-two";
        "₩",
        "displayName-count-few";
        "₩",
        "displayName-count-many";
        "₩",
        "displayName-count-other";
        "₩",
        "symbol";
        "₩",
        "symbol-alt-narrow";
        "₩";
    }
    "KWD";
    {
        "displayName";
        "دينار كويتي",
        "displayName-count-zero";
        "دينار كويتي",
        "displayName-count-one";
        "دينار كويتي",
        "displayName-count-two";
        "دينار كويتي",
        "displayName-count-few";
        "دينار كويتي",
        "displayName-count-many";
        "دينار كويتي",
        "displayName-count-other";
        "دينار كويتي",
        "symbol";
        "د.ك.";
    }
    "KYD";

```



```

{
    "displayName";
    "دولار جزر كيمن",
    "displayName-count-zero";
    "دولار جزر كيمن",
    "displayName-count-one";
    "دولار جزر كيمن",
    "displayName-count-two";
    "دولار جزر كيمن",
    "displayName-count-few";
    "دولار جزر كيمن",
    "displayName-count-many";
    "دولار جزر كيمن",
    "displayName-count-other";
    "دولار جزر كيمن",
    "symbol";
    "KYD",
    "symbol-alt-narrow";
    "KY$";
}
"KZT";
{
    "displayName";
    "تينغ كازاخستاني",
    "displayName-count-zero";
    "تينغ كازاخستاني",
    "displayName-count-one";
    "تينغ كازاخستاني",
    "displayName-count-two";
    "تينغ كازاخستاني",
    "displayName-count-few";
    "تينغ كازاخستاني",
    "displayName-count-many";
    "تينغ كازاخستاني",
    "displayName-count-other";
    "تينغ كازاخستاني",
    "symbol";
    "KZT",
    "symbol-alt-narrow";
    "Т";
}
"LAK";
{
    "displayName";
    "كيب لاوسي",
    "displayName-count-zero";
    "كيب لاوسي",
    "displayName-count-one";
    "كيب لاوسي",
    "displayName-count-two";
    "كيب لاوسي",
    "displayName-count-few";
    "كيب لاوسي",
    "displayName-count-many";
    "كيب لاوسي",
    "displayName-count-other";
    "كيب لاوسي";
}

```

```

        "symbol";
        "LAK",
        "symbol-alt-narrow";
        "₭";
    }
    "LBP";
    {
        "displayName";
        "جنيه لبناني",
        "displayName-count-zero";
        "جنيه لبناني",
        "displayName-count-one";
        "جنيه لبناني",
        "displayName-count-two";
        "جنيه لبناني",
        "displayName-count-few";
        "جنيه لبناني",
        "displayName-count-many";
        "جنيه لبناني",
        "displayName-count-other";
        "جنيه لبناني",
        "symbol";
        "ل.ل.",
        "symbol-alt-narrow";
        "L£";
    }
    "LKR";
    {
        "displayName";
        "روبية سريلانكية",
        "displayName-count-zero";
        "روبية سريلانكية",
        "displayName-count-one";
        "روبية سريلانكية",
        "displayName-count-two";
        "روبية سريلانكية",
        "displayName-count-few";
        "روبية سريلانكية",
        "displayName-count-many";
        "روبية سريلانكية",
        "displayName-count-other";
        "روبية سريلانكية",
        "symbol";
        "LKR",
        "symbol-alt-narrow";
        "Rs";
    }
    "LRD";
    {
        "displayName";
        "دولار ليبيري",
        "displayName-count-zero";
        "دولار ليبيري",
        "displayName-count-one";
        "دولار ليبيري",
        "displayName-count-two";
        "دولاران ليبيريان",

```

```

        "displayName-count-few";
        "دولارات ليبيرية",
        "displayName-count-many";
        "دولارًا ليبيريًا",
        "displayName-count-other";
        "دولار ليبيري",
        "symbol";
        "LRD",
        "symbol-alt-narrow";
        "$";
    }
    "LSL";
    {
        "displayName";
        "لوتي ليسوتو",
        "symbol";
        "LSL";
    }
    "LTL";
    {
        "displayName";
        "ليتلا ليتوانية",
        "displayName-count-zero";
        "ليتلا ليتوانية",
        "displayName-count-one";
        "ليتلا ليتوانية",
        "displayName-count-two";
        "ليتلا ليتوانية",
        "displayName-count-few";
        "ليتلا ليتوانية",
        "displayName-count-many";
        "ليتلا ليتوانية",
        "displayName-count-other";
        "ليتلا ليتوانية",
        "symbol";
        "LTL",
        "symbol-alt-narrow";
        "Lt";
    }
    "LTT";
    {
        "displayName";
        "تالوناس ليتواني",
        "symbol";
        "LTT";
    }
    "LUC";
    {
        "displayName";
        "فرنك لوكسمبرج قابل للتحويل",
        "symbol";
        "LUC";
    }
    "LUF";
    {
        "displayName";
        "فرنك لوكسمبرج",

```

```

        "symbol";
        "LUF";
    }
    "LUL";
    {
        "displayName";
        "فرنك لوكسمبرج المالي",
        "symbol";
        "LUL";
    }
    "LVL";
    {
        "displayName";
        "لاتس لاتفيا",
        "displayName-count-zero";
        "لاتس لاتفي",
        "displayName-count-one";
        "لاتس لاتفي",
        "displayName-count-two";
        "لاتس لاتفي",
        "displayName-count-few";
        "لاتس لاتفي",
        "displayName-count-many";
        "لاتس لاتفي",
        "displayName-count-other";
        "لاتس لاتفي",
        "symbol";
        "LVL",
        "symbol-alt-narrow";
        "Ls";
    }
    "LVR";
    {
        "displayName";
        "روبل لاتفيا",
        "symbol";
        "LVR";
    }
    "LYD";
    {
        "displayName";
        "دينار ليبي",
        "displayName-count-zero";
        "دينار ليبي",
        "displayName-count-one";
        "دينار ليبي",
        "displayName-count-two";
        "ديناران ليبيان",
        "displayName-count-few";
        "دينارات ليبية",
        "displayName-count-many";
        "دينارًا ليبيا",
        "displayName-count-other";
        "دينار ليبي",
        "symbol";
        "د.ل.";
    }
}

```

```

"MAD";
{
  "displayName";
  "درهم مغربي",
  "displayName-count-zero";
  "درهم مغربي",
  "displayName-count-one";
  "درهم مغربي",
  "displayName-count-two";
  "درهمان مغربيان",
  "displayName-count-few";
  "دراهم مغربية",
  "displayName-count-many";
  "درهما مغربيا",
  "displayName-count-other";
  "درهم مغربي",
  "symbol";
  "د.م.";
}
"MAF";
{
  "displayName";
  "فرنك مغربي",
  "symbol";
  "MAF";
}
"MCF";
{
  "displayName";
  "MCF",
  "symbol";
  "MCF";
}
"MDC";
{
  "displayName";
  "MDC",
  "symbol";
  "MDC";
}
"MDL";
{
  "displayName";
  "ليو مولدوفي",
  "displayName-count-zero";
  "ليو مولدوفي",
  "displayName-count-one";
  "ليو مولدوفي",
  "displayName-count-two";
  "ليو مولدوفي",
  "displayName-count-few";
  "ليو مولدوفي",
  "displayName-count-many";
  "ليو مولدوفي",
  "displayName-count-other";
  "ليو مولدوفي",
  "symbol";
}

```

```

        "MDL";
    }
    "MGA";
    {
        "displayName";
        "أرياري مدغشقر",
        "displayName-count-zero";
        "أرياري مدغشقر",
        "displayName-count-one";
        "أرياري مدغشقر",
        "displayName-count-two";
        "أرياري مدغشقر",
        "displayName-count-few";
        "أرياري مدغشقر",
        "displayName-count-many";
        "أرياري مدغشقر",
        "displayName-count-other";
        "أرياري مدغشقر",
        "symbol";
        "MGA",
        "symbol-alt-narrow";
        "Ar";
    }
    "MGF";
    {
        "displayName";
        "فرنك مدغشقر",
        "symbol";
        "MGF";
    }
    "MKD";
    {
        "displayName";
        "دينار مقدوني",
        "displayName-count-zero";
        "دينار مقدوني",
        "displayName-count-one";
        "دينار مقدوني",
        "displayName-count-two";
        "ديناران مقدونيان",
        "displayName-count-few";
        "دينارات مقدونية",
        "displayName-count-many";
        "دينارًا مقدونيا",
        "displayName-count-other";
        "دينار مقدوني",
        "symbol";
        "MKD";
    }
    "MKN";
    {
        "displayName";
        "MKN",
        "symbol";
        "MKN";
    }
    "MLF";

```

```

    {
      "displayName";
      "فرنك مالي",
      "symbol";
      "MLF";
    }
    "MMK";
    {
      "displayName";
      "كيات ميانمار",
      "displayName-count-zero";
      "كيات ميانمار",
      "displayName-count-one";
      "كيات ميانمار",
      "displayName-count-two";
      "كيات ميانمار",
      "displayName-count-few";
      "كيات ميانمار",
      "displayName-count-many";
      "كيات ميانمار",
      "displayName-count-other";
      "كيات ميانمار",
      "symbol";
      "MMK",
      "symbol-alt-narrow";
      "K";
    }
    "MNT";
    {
      "displayName";
      "توغروغ منغولي",
      "displayName-count-zero";
      "توغروغ منغولي",
      "displayName-count-one";
      "توغروغ منغولي",
      "displayName-count-two";
      "توغروغ منغولي",
      "displayName-count-few";
      "توغروغ منغولي",
      "displayName-count-many";
      "توغروغ منغولي",
      "displayName-count-other";
      "توغروغ منغولي",
      "symbol";
      "MNT",
      "symbol-alt-narrow";
      "₮";
    }
    "MOP";
    {
      "displayName";
      "باتاكا مكاوي",
      "displayName-count-zero";
      "باتاكا مكاوي",
      "displayName-count-one";
      "باتاكا مكاوي",
      "displayName-count-two";
    }
  }

```

```

        "باتاكا ماكاوي",
        "displayName-count-few";
        "باتاكا ماكاوي",
        "displayName-count-many";
        "باتاكا ماكاوي",
        "displayName-count-other";
        "باتاكا ماكاوي",
        "symbol";
        "MOP";
    }
    "MRO";
    {
        "displayName";
        "أوقية موريتانية",
        "displayName-count-zero";
        "أوقية موريتانية",
        "displayName-count-one";
        "أوقية موريتانية",
        "displayName-count-two";
        "أوقية موريتانية",
        "displayName-count-few";
        "أوقية موريتانية",
        "displayName-count-many";
        "أوقية موريتانية",
        "displayName-count-other";
        "أوقية موريتانية",
        "symbol";
        "أ.م.";
    }
    "MTL";
    {
        "displayName";
        "ليرة مالطية",
        "symbol";
        "MTL";
    }
    "MTP";
    {
        "displayName";
        "جنيه مالطي",
        "symbol";
        "MTP";
    }
    "MUR";
    {
        "displayName";
        "روبية موريشيوسية",
        "displayName-count-zero";
        "روبية موريشيوسية",
        "displayName-count-one";
        "روبية موريشيوسية",
        "displayName-count-two";
        "روبية موريشيوسية",
        "displayName-count-few";
        "روبية موريشيوسية",
        "displayName-count-many";
        "روبية موريشيوسية",
    }

```



```

        "displayName-count-other";
        "روبية موريشيوسية",
        "symbol";
        "MUR",
        "symbol-alt-narrow";
        "Rs";
    }
    "MVR";
    {
        "displayName";
        "روفيه جزر المالديف",
        "displayName-count-zero";
        "روفيه جزر المالديف",
        "displayName-count-one";
        "روفيه جزر المالديف",
        "displayName-count-two";
        "روفيه جزر المالديف",
        "displayName-count-few";
        "روفيه جزر المالديف",
        "displayName-count-many";
        "روفيه جزر المالديف",
        "displayName-count-other";
        "روفيه جزر المالديف",
        "symbol";
        "MVR";
    }
    "MWK";
    {
        "displayName";
        "كواشا مالاوي",
        "displayName-count-zero";
        "كواشا مالاوي",
        "displayName-count-one";
        "كواشا مالاوي",
        "displayName-count-two";
        "كواشا مالاوي",
        "displayName-count-few";
        "كواشا مالاوي",
        "displayName-count-many";
        "كواشا مالاوي",
        "displayName-count-other";
        "كواشا مالاوي",
        "symbol";
        "MWK";
    }
    "MXN";
    {
        "displayName";
        "بيزو مكسيكي",
        "displayName-count-zero";
        "بيزو مكسيكي",
        "displayName-count-one";
        "بيزو مكسيكي",
        "displayName-count-two";
        "بيزو مكسيكي",
        "displayName-count-few";
        "بيزو مكسيكي",
    }

```

```

        "displayName-count-many";
        "بیزو مکسیکی",
        "displayName-count-other";
        "بیزو مکسیکی",
        "symbol";
        "MX$";
        "symbol-alt-narrow";
        "MX$";
    }
    "MXP";
    {
        "displayName";
        "1992-1861 - بیزو فضی مکسیکی",
        "symbol";
        "MXP";
    }
    "MXV";
    {
        "displayName";
        "MXV",
        "symbol";
        "MXV";
    }
    "MYR";
    {
        "displayName";
        "رینگیت مالیزی",
        "displayName-count-zero";
        "رینگیت مالیزی",
        "displayName-count-one";
        "رینگیت مالیزی",
        "displayName-count-two";
        "رینگیت مالیزی",
        "displayName-count-few";
        "رینگیت مالیزی",
        "displayName-count-many";
        "رینگیت مالیزی",
        "displayName-count-other";
        "رینگیت مالیزی",
        "symbol";
        "MYR",
        "symbol-alt-narrow";
        "RM";
    }
    "MZE";
    {
        "displayName";
        "اسکود موزمبیقی",
        "symbol";
        "MZE";
    }
    "MZM";
    {
        "displayName";
        "MZM",
        "symbol";
        "MZM";
    }

```

```

    }
    "MZN";
    {
        "displayName";
        "متكال موزمبيقى",
        "displayName-count-zero";
        "متكال موزمبيقى",
        "displayName-count-one";
        "متكال موزمبيقى",
        "displayName-count-two";
        "متكال موزمبيقى",
        "displayName-count-few";
        "متكال موزمبيقى",
        "displayName-count-many";
        "متكال موزمبيقى",
        "displayName-count-other";
        "متكال موزمبيقى",
        "symbol";
        "MZN";
    }
    "NAD";
    {
        "displayName";
        "دولار نامىبى",
        "displayName-count-zero";
        "دولار نامىبى",
        "displayName-count-one";
        "دولار نامىبى",
        "displayName-count-two";
        "دولار نامىبى",
        "displayName-count-few";
        "دولار نامىبى",
        "displayName-count-many";
        "دولار نامىبى",
        "displayName-count-other";
        "دولار نامىبى",
        "symbol";
        "NAD",
        "symbol-alt-narrow";
        "$";
    }
    "NGN";
    {
        "displayName";
        "نايرا نيجيرى",
        "displayName-count-zero";
        "نايرا نيجيرى",
        "displayName-count-one";
        "نايرا نيجيرى",
        "displayName-count-two";
        "نايرا نيجيرى",
        "displayName-count-few";
        "نايرا نيجيرى",
        "displayName-count-many";
        "نايرا نيجيرى",
        "displayName-count-other";
        "نايرا نيجيرى",
    }

```

```

        "symbol";
        "NGN",
        "symbol-alt-narrow";
        "₦";
    }
    "NIC";
    {
        "displayName";
        "كوردوبه نيكاراغوا",
        "symbol";
        "NIC";
    }
    "NIO";
    {
        "displayName";
        "قرطبة نيكاراغوا",
        "displayName-count-zero";
        "قرطبة نيكاراغوا",
        "displayName-count-one";
        "قرطبة نيكاراغوا",
        "displayName-count-two";
        "قرطبة نيكاراغوا",
        "displayName-count-few";
        "قرطبة نيكاراغوا",
        "displayName-count-many";
        "قرطبة نيكاراغوا",
        "displayName-count-other";
        "قرطبة نيكاراغوا",
        "symbol";
        "NIO",
        "symbol-alt-narrow";
        "C$";
    }
    "NLG";
    {
        "displayName";
        "جلدر هولندي",
        "symbol";
        "NLG";
    }
    "NOK";
    {
        "displayName";
        "كرونة نرويجية",
        "displayName-count-zero";
        "كرونة نرويجية",
        "displayName-count-one";
        "كرونة نرويجية",
        "displayName-count-two";
        "كرونة نرويجية",
        "displayName-count-few";
        "كرونة نرويجية",
        "displayName-count-many";
        "كرونة نرويجية",
        "displayName-count-other";
        "كرونة نرويجية",
        "symbol";
    }

```

```

        "NOK",
        "symbol-alt-narrow";
        "kr";
    }
    "NPR";
    {
        "displayName";
        "روبية نيبالي",
        "displayName-count-zero";
        "روبية نيبالي",
        "displayName-count-one";
        "روبية نيبالي",
        "displayName-count-two";
        "روبية نيبالي",
        "displayName-count-few";
        "روبية نيبالي",
        "displayName-count-many";
        "روبية نيبالي",
        "displayName-count-other";
        "روبية نيبالي",
        "symbol";
        "NPR",
        "symbol-alt-narrow";
        "Rs";
    }
    "NZD";
    {
        "displayName";
        "دولار نيوزيلندي",
        "displayName-count-zero";
        "دولار نيوزيلندي",
        "displayName-count-one";
        "دولار نيوزيلندي",
        "displayName-count-two";
        "دولار نيوزيلندي",
        "displayName-count-few";
        "دولار نيوزيلندي",
        "displayName-count-many";
        "دولار نيوزيلندي",
        "displayName-count-other";
        "دولار نيوزيلندي",
        "symbol";
        "NZ$",
        "symbol-alt-narrow";
        "NZ$";
    }
    "OMR";
    {
        "displayName";
        "رول عماني",
        "displayName-count-zero";
        "رول عماني",
        "displayName-count-one";
        "رول عماني",
        "displayName-count-two";
        "رول عماني",
        "displayName-count-few";
    }

```

```

        "ول عماني",
        "displayName-count-many";
        "ول عماني",
        "displayName-count-other";
        "ول عماني",
        "symbol";
        "ر.ع.";
    }
    "PAB";
    {
        "displayName";
        "بالبوا بنمي",
        "displayName-count-zero";
        "بالبوا بنمي",
        "displayName-count-one";
        "بالبوا بنمي",
        "displayName-count-two";
        "بالبوا بنمي",
        "displayName-count-few";
        "بالبوا بنمي",
        "displayName-count-many";
        "بالبوا بنمي",
        "displayName-count-other";
        "بالبوا بنمي",
        "symbol";
        "PAB";
    }
    "PEI";
    {
        "displayName";
        "PEI",
        "symbol";
        "PEI";
    }
    "PEN";
    {
        "displayName";
        "سول جديد البيرو",
        "displayName-count-zero";
        "سول جديد البيرو",
        "displayName-count-one";
        "سول جديد البيرو",
        "displayName-count-two";
        "سول جديد البيرو",
        "displayName-count-few";
        "سول جديد البيرو",
        "displayName-count-many";
        "سول جديد البيرو",
        "displayName-count-other";
        "سول جديد البيرو",
        "symbol";
        "PEN";
    }
    "PES";
    {
        "displayName";
        "PES",

```

```

        "symbol";
        "PES";
    }
    "PGK";
    {
        "displayName";
        "كيننا بابوا غينيا الجديدة",
        "displayName-count-zero";
        "كيننا بابوا غينيا الجديدة",
        "displayName-count-one";
        "كيننا بابوا غينيا الجديدة",
        "displayName-count-two";
        "كيننا بابوا غينيا الجديدة",
        "displayName-count-few";
        "كيننا بابوا غينيا الجديدة",
        "displayName-count-many";
        "كيننا بابوا غينيا الجديدة",
        "displayName-count-other";
        "كيننا بابوا غينيا الجديدة",
        "symbol";
        "PGK";
    }
    "PHP";
    {
        "displayName";
        "بيزو فلبيني",
        "displayName-count-zero";
        "بيزو فلبيني",
        "displayName-count-one";
        "بيزو فلبيني",
        "displayName-count-two";
        "بيزو فلبيني",
        "displayName-count-few";
        "بيزو فلبيني",
        "displayName-count-many";
        "بيزو فلبيني",
        "displayName-count-other";
        "بيزو فلبيني",
        "symbol";
        "PHP",
        "symbol-alt-narrow";
        "₱";
    }
    "PKR";
    {
        "displayName";
        "روبية باكستاني",
        "displayName-count-zero";
        "روبية باكستاني",
        "displayName-count-one";
        "روبية باكستاني",
        "displayName-count-two";
        "روبية باكستاني",
        "displayName-count-few";
        "روبية باكستاني",
        "displayName-count-many";
        "روبية باكستاني",
    }

```

```

        "displayName-count-other";
        "روبيۃ پاڪستاني",
        "symbol";
        "PKR",
        "symbol-alt-narrow";
        "Rs";
    }
    "PLN";
    {
        "displayName";
        "زلوتي پولندي",
        "displayName-count-zero";
        "زلوتي پولندي",
        "displayName-count-one";
        "زلوتي پولندي",
        "displayName-count-two";
        "زلوتي پولندي",
        "displayName-count-few";
        "زلوتي پولندي",
        "displayName-count-many";
        "زلوتي پولندي",
        "displayName-count-other";
        "زلوتي پولندي",
        "symbol";
        "PLN",
        "symbol-alt-narrow";
        "zł";
    }
    "PLZ";
    {
        "displayName";
        "زلوتي پولندي - 1950-1995",
        "symbol";
        "PLZ";
    }
    "PTE";
    {
        "displayName";
        "اسڪود پرتگالي",
        "symbol";
        "PTE";
    }
    "PYG";
    {
        "displayName";
        "جواراني پاراگوای",
        "displayName-count-zero";
        "جواراني پاراگوای",
        "displayName-count-one";
        "جواراني پاراگوای",
        "displayName-count-two";
        "جواراني پاراگوای",
        "displayName-count-few";
        "جواراني پاراگوای",
        "displayName-count-many";
        "جواراني پاراگوای",
        "displayName-count-other";
    }

```



```

        "جواراني باراجوای",
        "symbol";
    "PYG",
        "symbol-alt-narrow";
    "₲";
}
"QAR";
{
    "displayName";
    "قطري",
        "displayName-count-zero";
    "قطري",
        "displayName-count-one";
    "قطري",
        "displayName-count-two";
    "قطري",
        "displayName-count-few";
    "قطري",
        "displayName-count-many";
    "قطري",
        "displayName-count-other";
    "قطري",
        "symbol";
    "ر.ق.";
}
"RHD";
{
    "displayName";
    "دولار رودیسی",
        "symbol";
    "RHD";
}
"ROL";
{
    "displayName";
    "لیو رومانی قدیم",
        "symbol";
    "ROL";
}
"RON";
{
    "displayName";
    "لیو رومانی",
        "displayName-count-zero";
    "لیو رومانی",
        "displayName-count-one";
    "لیو رومانی",
        "displayName-count-two";
    "لیو رومانی",
        "displayName-count-few";
    "لیو رومانی",
        "displayName-count-many";
    "لیو رومانی",
        "displayName-count-other";
    "لیو رومانی",
        "symbol";
    "RON";
}

```

```

        "symbol-alt-narrow";
        "lei";
    }
    "RSD";
    {
        "displayName";
        "دينار صربي",
        "displayName-count-zero";
        "دينار صربي",
        "displayName-count-one";
        "دينار صربي",
        "displayName-count-two";
        "ديناران صربيان",
        "displayName-count-few";
        "دينارات صربية",
        "displayName-count-many";
        "دينارًا صربيًا",
        "displayName-count-other";
        "دينار صربي",
        "symbol";
        "RSD";
    }
    "RUB";
    {
        "displayName";
        "روبل روسي",
        "displayName-count-zero";
        "روبل روسي",
        "displayName-count-one";
        "روبل روسي",
        "displayName-count-two";
        "روبل روسي",
        "displayName-count-few";
        "روبل روسي",
        "displayName-count-many";
        "روبل روسي",
        "displayName-count-other";
        "روبل روسي",
        "symbol";
        "RUB",
        "symbol-alt-narrow";
        "₽";
    }
    "RUR";
    {
        "displayName";
        "1998-1991 - روبل روسي",
        "symbol";
        "RUR",
        "symbol-alt-narrow";
        "p.";
    }
    "RWF";
    {
        "displayName";
        "فرنك رواندي",
        "displayName-count-zero";

```

```

        "فرنك رواندي",
        "displayName-count-one";
        "فرنك رواندي",
        "displayName-count-two";
        "فرنك رواندي",
        "displayName-count-few";
        "فرنك رواندي",
        "displayName-count-many";
        "فرنك رواندي",
        "displayName-count-other";
        "فرنك رواندي",
        "symbol";
        "RWF",
        "symbol-alt-narrow";
        "RF";
    }
    "SAR";
    {
        "displayName";
        "رول سعودي",
        "displayName-count-zero";
        "رول سعودي",
        "displayName-count-one";
        "رول سعودي",
        "displayName-count-two";
        "رول سعودي",
        "displayName-count-few";
        "رول سعودي",
        "displayName-count-many";
        "رول سعودي",
        "displayName-count-other";
        "رول سعودي",
        "symbol";
        "ر.س.";
    }
    "SBD";
    {
        "displayName";
        "دولار جزر سليمان",
        "displayName-count-zero";
        "دولار جزر سليمان",
        "displayName-count-one";
        "دولار جزر سليمان",
        "displayName-count-two";
        "دولار جزر سليمان",
        "displayName-count-few";
        "دولار جزر سليمان",
        "displayName-count-many";
        "دولار جزر سليمان",
        "displayName-count-other";
        "دولار جزر سليمان",
        "symbol";
        "SBD",
        "symbol-alt-narrow";
        "SB$";
    }
    "SCR";

```

```

{
  "displayName";
  "روبية سيشيلية",
  "displayName-count-zero";
  "روبية سيشيلية",
  "displayName-count-one";
  "روبية سيشيلية",
  "displayName-count-two";
  "روبية سيشيلية",
  "displayName-count-few";
  "روبية سيشيلية",
  "displayName-count-many";
  "روبية سيشيلية",
  "displayName-count-other";
  "روبية سيشيلية",
  "symbol";
  "SCR";
}
"SDD";
{
  "displayName";
  "دينار سوداني",
  "symbol";
  "د.س.";
}
"SDG";
{
  "displayName";
  "جنيه سوداني",
  "displayName-count-zero";
  "جنيه سوداني",
  "displayName-count-one";
  "جنيه سوداني",
  "displayName-count-two";
  "جنيه سوداني",
  "displayName-count-few";
  "جنيهات سودانية",
  "displayName-count-many";
  "جنيهاً سودانياً",
  "displayName-count-other";
  "جنيه سوداني",
  "symbol";
  "ج.س.";
}
"SDP";
{
  "displayName";
  "جنيه سوداني قديم",
  "symbol";
  "SDP";
}
"SEK";
{
  "displayName";
  "كرونة سويدية",
  "displayName-count-zero";
  "كرونة سويدية",

```

```

        "displayName-count-one";
        "كرونة سويدية",
        "displayName-count-two";
        "كرونة سويدية",
        "displayName-count-few";
        "كرونة سويدية",
        "displayName-count-many";
        "كرونة سويدية",
        "displayName-count-other";
        "كرونة سويدية",
        "symbol";
        "SEK",
        "symbol-alt-narrow";
        "kr";
    }
    "SGD";
    {
        "displayName";
        "دولار سنغافوري",
        "displayName-count-zero";
        "دولار سنغافوري",
        "displayName-count-one";
        "دولار سنغافوري",
        "displayName-count-two";
        "دولار سنغافوري",
        "displayName-count-few";
        "دولار سنغافوري",
        "displayName-count-many";
        "دولار سنغافوري",
        "displayName-count-other";
        "دولار سنغافوري",
        "symbol";
        "SGD",
        "symbol-alt-narrow";
        "$";
    }
    "SHP";
    {
        "displayName";
        "جنيه سانت هيلين",
        "displayName-count-zero";
        "جنيه سانت هيلين",
        "displayName-count-one";
        "جنيه سانت هيلين",
        "displayName-count-two";
        "جنيه سانت هيلين",
        "displayName-count-few";
        "جنيه سانت هيلين",
        "displayName-count-many";
        "جنيه سانت هيلين",
        "displayName-count-other";
        "جنيه سانت هيلين",
        "symbol";
        "SHP",
        "symbol-alt-narrow";
        "£";
    }
}

```

```

    "SIT";
    {
        "displayName";
        "تولار سلوفيني",
        "symbol";
        "SIT";
    }
    "SKK";
    {
        "displayName";
        "كرونه سلوفاكية",
        "symbol";
        "SKK";
    }
    "SLL";
    {
        "displayName";
        "ليون سيراليوني",
        "displayName-count-zero";
        "ليون سيراليوني",
        "displayName-count-one";
        "ليون سيراليوني",
        "displayName-count-two";
        "ليون سيراليوني",
        "displayName-count-few";
        "ليون سيراليوني",
        "displayName-count-many";
        "ليون سيراليوني",
        "displayName-count-other";
        "ليون سيراليوني",
        "symbol";
        "SLL";
    }
    "SOS";
    {
        "displayName";
        "شلم صومالي",
        "displayName-count-zero";
        "شلم صومالي",
        "displayName-count-one";
        "شلم صومالي",
        "displayName-count-two";
        "شلم صومالي",
        "displayName-count-few";
        "شلم صومالي",
        "displayName-count-many";
        "شلم صومالي",
        "displayName-count-other";
        "شلم صومالي",
        "symbol";
        "SOS";
    }
    "SRD";
    {
        "displayName";
        "دولار سورينامي",
        "displayName-count-zero";

```

```

        "دولار سورينامي",
        "displayName-count-one";
        "دولار سورينامي",
        "displayName-count-two";
        "دولار سورينامي",
        "displayName-count-few";
        "دولار سورينامي",
        "displayName-count-many";
        "دولار سورينامي",
        "displayName-count-other";
        "دولار سورينامي",
        "symbol";
        "SRD",
        "symbol-alt-narrow";
        "SR$";
    }
    "SRG";
    {
        "displayName";
        "جلدر سورينامي",
        "symbol";
        "SRG";
    }
    "SSP";
    {
        "displayName";
        "جنيه جنوب السودان",
        "displayName-count-zero";
        "جنيه جنوب السودان",
        "displayName-count-one";
        "جنيه جنوب السودان",
        "displayName-count-two";
        "جنيهان جنوب السودان",
        "displayName-count-few";
        "جنيهات جنوب السودان",
        "displayName-count-many";
        "جنيها جنوب السودان",
        "displayName-count-other";
        "جنيه جنوب السودان",
        "symbol";
        "SSP",
        "symbol-alt-narrow";
        "£";
    }
    "STD";
    {
        "displayName";
        "دوبرا ساو تومي وبرينسيبي",
        "displayName-count-zero";
        "دوبرا ساو تومي وبرينسيبي",
        "displayName-count-one";
        "دوبرا ساو تومي وبرينسيبي",
        "displayName-count-two";
        "دوبرا ساو تومي وبرينسيبي",
        "displayName-count-few";
        "دوبرا ساو تومي وبرينسيبي",
        "displayName-count-many";
    }

```

```

        "دوبرا ساو تومي وبرينسيبي",
        "displayName-count-other";
        "دوبرا ساو تومي وبرينسيبي",
        "symbol";
        "STD",
        "symbol-alt-narrow";
        "Db";
    }
    "SUR";
    {
        "displayName";
        "روبل سوفيتي",
        "symbol";
        "SUR";
    }
    "SVC";
    {
        "displayName";
        "كولون سلفادوري",
        "symbol";
        "SVC";
    }
    "SYP";
    {
        "displayName";
        "ليرة سورية",
        "displayName-count-zero";
        "ليرة سورية",
        "displayName-count-one";
        "ليرة سورية",
        "displayName-count-two";
        "ليرة سورية",
        "displayName-count-few";
        "ليرة سورية",
        "displayName-count-many";
        "ليرة سورية",
        "displayName-count-other";
        "ليرة سورية",
        "symbol";
        "ل.س.",
        "symbol-alt-narrow";
        "£";
    }
    "SZL";
    {
        "displayName";
        "ليلانجيني سوازيلندي",
        "displayName-count-zero";
        "ليلانجيني سوازيلندي",
        "displayName-count-one";
        "ليلانجيني سوازيلندي",
        "displayName-count-two";
        "ليلانجيني سوازيلندي",
        "displayName-count-few";
        "ليلانجيني سوازيلندي",
        "displayName-count-many";
        "ليلانجيني سوازيلندي",
    }

```



```

        "displayName-count-other";
        "ليلانجيني سوازيلندي",
        "symbol";
        "SZL";
    }
    "THB";
    {
        "displayName";
        "باخت تايلاندي",
        "displayName-count-zero";
        "باخت تايلاندي",
        "displayName-count-one";
        "باخت تايلاندي",
        "displayName-count-two";
        "باخت تايلاندي",
        "displayName-count-few";
        "باخت تايلاندي",
        "displayName-count-many";
        "باخت تايلاندي",
        "displayName-count-other";
        "باخت تايلاندي",
        "symbol";
        "฿",
        "symbol-alt-narrow";
        "฿";
    }
    "TJR";
    {
        "displayName";
        "روبل طاجيكستاني",
        "symbol";
        "TJR";
    }
    "TJS";
    {
        "displayName";
        "سوموني طاجيكستاني",
        "displayName-count-zero";
        "سوموني طاجيكستاني",
        "displayName-count-one";
        "سوموني طاجيكستاني",
        "displayName-count-two";
        "سوموني طاجيكستاني",
        "displayName-count-few";
        "سوموني طاجيكستاني",
        "displayName-count-many";
        "سوموني طاجيكستاني",
        "displayName-count-other";
        "سوموني طاجيكستاني",
        "symbol";
        "TJS";
    }
    "TMM";
    {
        "displayName";
        "مانات تركمنستاني",
        "symbol";

```

```

        "TMM";
    }
    "TMT";
    {
        "displayName";
        "مانات ترکمانستان",
        "displayName-count-zero";
        "مانات ترکمانستان",
        "displayName-count-one";
        "مانات ترکمانستان",
        "displayName-count-two";
        "مانات ترکمانستان",
        "displayName-count-few";
        "مانات ترکمانستان",
        "displayName-count-many";
        "مانات ترکمانستان",
        "displayName-count-other";
        "مانات ترکمانستان",
        "symbol";
        "TMT";
    }
    "TND";
    {
        "displayName";
        "دينار تونسي",
        "displayName-count-zero";
        "دينار تونسي",
        "displayName-count-one";
        "دينار تونسي",
        "displayName-count-two";
        "ديناران تونسيان",
        "displayName-count-few";
        "دينارات تونسية",
        "displayName-count-many";
        "دينارًا تونسيًا",
        "displayName-count-other";
        "دينار تونسي",
        "symbol";
        "د.ت.";
    }
    "TOP";
    {
        "displayName";
        "بانغا تونغا",
        "displayName-count-zero";
        "بانغا تونغا",
        "displayName-count-one";
        "بانغا تونغا",
        "displayName-count-two";
        "بانغا تونغا",
        "displayName-count-few";
        "بانغا تونغا",
        "displayName-count-many";
        "بانغا تونغا",
        "displayName-count-other";
        "بانغا تونغا",
        "symbol";
    }

```

```

        "TOP",
        "symbol-alt-narrow";
        "T$";
    }
    "TPE";
    {
        "displayName";
        "اسكود تيموري",
        "symbol";
        "TPE";
    }
    "TRL";
    {
        "displayName";
        "ليرة تركي",
        "symbol";
        "TRL";
    }
    "TRY";
    {
        "displayName";
        "ليرة تركية",
        "displayName-count-zero";
        "ليرة تركية",
        "displayName-count-one";
        "ليرة تركية",
        "displayName-count-two";
        "ليرة تركية",
        "displayName-count-few";
        "ليرة تركية",
        "displayName-count-many";
        "ليرة تركية",
        "displayName-count-other";
        "ليرة تركية",
        "symbol";
        "TRY",
        "symbol-alt-narrow";
        "₺",
        "symbol-alt-variant";
        "TL";
    }
    "TTD";
    {
        "displayName";
        "دولار ترينداد وتوباغو",
        "displayName-count-zero";
        "دولار ترينداد وتوباغو",
        "displayName-count-one";
        "دولار ترينداد وتوباغو",
        "displayName-count-two";
        "دولار ترينداد وتوباغو",
        "displayName-count-few";
        "دولار ترينداد وتوباغو",
        "displayName-count-many";
        "دولار ترينداد وتوباغو",
        "displayName-count-other";
        "دولار ترينداد وتوباغو",
    }

```

```

        "symbol";
        "TTD",
        "symbol-alt-narrow";
        "TT$";
    }
    "TWD";
    {
        "displayName";
        "دولار تایوانی",
        "displayName-count-zero";
        "دولار تایوانی",
        "displayName-count-one";
        "دولار تایوانی",
        "displayName-count-two";
        "دولار تایوانی",
        "displayName-count-few";
        "دولار تایوانی",
        "displayName-count-many";
        "دولار تایوانی",
        "displayName-count-other";
        "دولار تایوانی",
        "symbol";
        "NT$";
        "symbol-alt-narrow";
        "NT$";
    }
    "TZS";
    {
        "displayName";
        "شلن تنزانی",
        "displayName-count-zero";
        "شلن تنزانی",
        "displayName-count-one";
        "شلن تنزانی",
        "displayName-count-two";
        "شلن تنزانی",
        "displayName-count-few";
        "شلن تنزانی",
        "displayName-count-many";
        "شلن تنزانی",
        "displayName-count-other";
        "شلن تنزانی",
        "symbol";
        "TZS";
    }
    "UAH";
    {
        "displayName";
        "هریفنیا اوکراینی",
        "displayName-count-zero";
        "هریفنیا اوکراینی",
        "displayName-count-one";
        "هریفنیا اوکراینی",
        "displayName-count-two";
        "هریفنیا اوکراینی",
        "displayName-count-few";
        "هریفنیا اوکراینی",
    }

```

```

        "displayName-count-many";
        "هريفنيا أوكراڻي",
        "displayName-count-other";
        "هريفنيا أوكراڻي",
        "symbol";
        "UAH",
        "symbol-alt-narrow";
        "₴";
    }
    "UAK";
    {
        "displayName";
        "UAK",
        "symbol";
        "UAK";
    }
    "UGS";
    {
        "displayName";
        "شلن أوغندي - 1966-1987",
        "symbol";
        "UGS";
    }
    "UGX";
    {
        "displayName";
        "شلن أوغندي",
        "displayName-count-zero";
        "شلن أوغندي",
        "displayName-count-one";
        "شلن أوغندي",
        "displayName-count-two";
        "شلن أوغندي",
        "displayName-count-few";
        "شلن أوغندي",
        "displayName-count-many";
        "شلن أوغندي",
        "displayName-count-other";
        "شلن أوغندي",
        "symbol";
        "UGX";
    }
    "USD";
    {
        "displayName";
        "دولار أمريكي",
        "displayName-count-zero";
        "دولار أمريكي",
        "displayName-count-one";
        "دولار أمريكي",
        "displayName-count-two";
        "دولار أمريكي",
        "displayName-count-few";
        "دولار أمريكي",
        "displayName-count-many";
        "دولار أمريكي",
        "displayName-count-other";
    }

```

```

        "دولار أمريكي",
        "symbol";
        "US$",
        "symbol-alt-narrow";
        "US$";
    }
    "USN";
    {
        "displayName";
        "دولار أمريكي (اليوم التالي)",
        "symbol";
        "USN";
    }
    "USS";
    {
        "displayName";
        "دولار أمريكي (نفس اليوم)",
        "symbol";
        "USS";
    }
    "UYI";
    {
        "displayName";
        "UYI",
        "symbol";
        "UYI";
    }
    "UYP";
    {
        "displayName";
        "بیزو اوروغواي - 1993-1975",
        "symbol";
        "UYP";
    }
    "UYU";
    {
        "displayName";
        "بیزو اوروغواي",
        "displayName-count-zero";
        "بیزو اوروغواي",
        "displayName-count-one";
        "بیزو اوروغواي",
        "displayName-count-two";
        "بیزو اوروغواي",
        "displayName-count-few";
        "بیزو اوروغواي",
        "displayName-count-many";
        "بیزو اوروغواي",
        "displayName-count-other";
        "بیزو اوروغواي",
        "symbol";
        "UYU",
        "symbol-alt-narrow";
        "UY$";
    }
    "UZS";
    {

```

```

        "displayName";
        "سوم أوزبكستاني",
        "displayName-count-zero";
        "سوم أوزبكستاني",
        "displayName-count-one";
        "سوم أوزبكستاني",
        "displayName-count-two";
        "سوم أوزبكستاني",
        "displayName-count-few";
        "سوم أوزبكستاني",
        "displayName-count-many";
        "سوم أوزبكستاني",
        "displayName-count-other";
        "سوم أوزبكستاني",
        "symbol";
        "UZS";
    }
    "VEB";
    {
        "displayName";
        "بوليفار فنزويلي - 1871-2008",
        "symbol";
        "VEB";
    }
    "VEF";
    {
        "displayName";
        "بوليفار فنزويلي",
        "displayName-count-zero";
        "بوليفار فنزويلي",
        "displayName-count-one";
        "بوليفار فنزويلي",
        "displayName-count-two";
        "بوليفار فنزويلي",
        "displayName-count-few";
        "بوليفار فنزويلي",
        "displayName-count-many";
        "بوليفار فنزويلي",
        "displayName-count-other";
        "بوليفار فنزويلي",
        "symbol";
        "VEF",
        "symbol-alt-narrow";
        "Bs";
    }
    "VND";
    {
        "displayName";
        "دونج فيتنامي",
        "displayName-count-zero";
        "دونج فيتنامي",
        "displayName-count-one";
        "دونج فيتنامي",
        "displayName-count-two";
        "دونج فيتنامي",
        "displayName-count-few";
        "دونج فيتنامي",
    }

```

```

        "displayName-count-many";
        "دونج فيتنامي",
        "displayName-count-other";
        "دونج فيتنامي",
        "symbol";
        "₫",
        "symbol-alt-narrow";
        "₫";
    }
    "VNN";
    {
        "displayName";
        "VNN",
        "symbol";
        "VNN";
    }
    "VUV";
    {
        "displayName";
        "فاتو فانواتو",
        "displayName-count-zero";
        "فاتو فانواتو",
        "displayName-count-one";
        "فاتو فانواتو",
        "displayName-count-two";
        "فاتو فانواتو",
        "displayName-count-few";
        "فاتو فانواتو",
        "displayName-count-many";
        "فاتو فانواتو",
        "displayName-count-other";
        "فاتو فانواتو",
        "symbol";
        "VUV";
    }
    "WST";
    {
        "displayName";
        "تالا ساموا",
        "displayName-count-zero";
        "تالا ساموا",
        "displayName-count-one";
        "تالا ساموا",
        "displayName-count-two";
        "تالا ساموا",
        "displayName-count-few";
        "تالا ساموا",
        "displayName-count-many";
        "تالا ساموا",
        "displayName-count-other";
        "تالا ساموا",
        "symbol";
        "WST";
    }
    "XAF";
    {
        "displayName";

```



```

        "فرنك وسط أفريقي",
        "displayName-count-zero";
        "فرنك وسط أفريقي",
        "displayName-count-one";
        "فرنك وسط أفريقي",
        "displayName-count-two";
        "فرنك وسط أفريقي",
        "displayName-count-few";
        "فرنك وسط أفريقي",
        "displayName-count-many";
        "فرنك وسط أفريقي",
        "displayName-count-other";
        "فرنك وسط أفريقي",
        "symbol";
        "FCFA";
    }
    "XAG";
    {
        "displayName";
        "فضة",
        "symbol";
        "XAG";
    }
    "XAU";
    {
        "displayName";
        "ذهب",
        "symbol";
        "XAU";
    }
    "XBA";
    {
        "displayName";
        "الوحدة الأوروبية المركبة",
        "symbol";
        "XBA";
    }
    "XBB";
    {
        "displayName";
        "الوحدة المالية الأوروبية",
        "symbol";
        "XBB";
    }
    "XBC";
    {
        "displayName";
        "الوحدة الحسابية الأوروبية",
        "symbol";
        "XBC";
    }
    "XBD";
    {
        "displayName";
        "وحدة الحساب الأوروبية (XBD)",
        "symbol";
        "XBD";
    }

```

```

    }
    "XCD";
    {
        "displayName";
        "دولار شرق الكاريبي",
        "displayName-count-zero";
        "دولار شرق الكاريبي",
        "displayName-count-one";
        "دولار شرق الكاريبي",
        "displayName-count-two";
        "دولار شرق الكاريبي",
        "displayName-count-few";
        "دولار شرق الكاريبي",
        "displayName-count-many";
        "دولار شرق الكاريبي",
        "displayName-count-other";
        "دولار شرق الكاريبي",
        "symbol";
        "EC$";
        "symbol-alt-narrow";
        "$";
    }
    "XDR";
    {
        "displayName";
        "حقوق السحب الخاصة",
        "symbol";
        "XDR";
    }
    "XEU";
    {
        "displayName";
        "وحدة النقد الأوروبية",
        "symbol";
        "XEU";
    }
    "XFO";
    {
        "displayName";
        "فرنك فرنسي ذهبي",
        "symbol";
        "XFO";
    }
    "XFU";
    {
        "displayName";
        "فرنك فرنسي (UIC)",
        "symbol";
        "XFU";
    }
    "XOF";
    {
        "displayName";
        "فرنك غرب أفريقي",
        "displayName-count-zero";
        "فرنك غرب أفريقي",
        "displayName-count-one";
    }

```

```

        "فرنك غرب أفريقي",
        "displayName-count-two";
        "فرنك غرب أفريقي",
        "displayName-count-few";
        "فرنك غرب أفريقي",
        "displayName-count-many";
        "فرنك غرب أفريقي",
        "displayName-count-other";
        "فرنك غرب أفريقي",
        "symbol";
        "CFA";
    }
    "XPD";
    {
        "displayName";
        "بالاديوم",
        "symbol";
        "XPD";
    }
    "XPF";
    {
        "displayName";
        "فرنك سي إف بي",
        "displayName-count-zero";
        "فرنك سي إف بي",
        "displayName-count-one";
        "فرنك سي إف بي",
        "displayName-count-two";
        "فرنك سي إف بي",
        "displayName-count-few";
        "فرنك سي إف بي",
        "displayName-count-many";
        "فرنك سي إف بي",
        "displayName-count-other";
        "فرنك سي إف بي",
        "symbol";
        "CFPF";
    }
    "XPT";
    {
        "displayName";
        "البلاطين",
        "symbol";
        "XPT";
    }
    "XRE";
    {
        "displayName";
        "XRE",
        "symbol";
        "XRE";
    }
    "XSU";
    {
        "displayName";
        "XSU",
        "symbol";
    }

```

```

        "XSU";
    }
    "XTS";
    {
        "displayName";
        "كود اختبار العملة",
        "symbol";
        "XTS";
    }
    "XUA";
    {
        "displayName";
        "XUA",
        "symbol";
        "XUA";
    }
    "XXX";
    {
        "displayName";
        "عملة غير معروفة",
        "displayName-count-zero";
        "(عملة غير معروفة)",
        "displayName-count-one";
        "(عملة غير معروفة)",
        "displayName-count-two";
        "(عملة غير معروفة)",
        "displayName-count-few";
        "(عملة غير معروفة)",
        "displayName-count-many";
        "(عملة غير معروفة)",
        "displayName-count-other";
        "(عملة غير معروفة)",
        "symbol";
        "***";
    }
    "YDD";
    {
        "displayName";
        "دينار يمني",
        "symbol";
        "YDD";
    }
    "YER";
    {
        "displayName";
        "رل يمني",
        "displayName-count-zero";
        "رل يمني",
        "displayName-count-one";
        "رل يمني",
        "displayName-count-two";
        "رل يمني",
        "displayName-count-few";
        "رل يمني",
        "displayName-count-many";
        "رل يمني",
        "displayName-count-other";
    }

```

```

        "دول يمني",
        "symbol";
        "ر.ي.";
    }
    "YUD";
    {
        "displayName";
        "دينار يوغسلافي",
        "symbol";
        "YUD";
    }
    "YUM";
    {
        "displayName";
        "YUM",
        "symbol";
        "YUM";
    }
    "YUN";
    {
        "displayName";
        "دينار يوغسلافي قابل للتحويل",
        "symbol";
        "YUN";
    }
    "YUR";
    {
        "displayName";
        "YUR",
        "symbol";
        "YUR";
    }
    "ZAL";
    {
        "displayName";
        "راند جنوب أفريقيا -مالي",
        "symbol";
        "ZAL";
    }
    "ZAR";
    {
        "displayName";
        "راند جنوب أفريقيا",
        "displayName-count-zero";
        "راند جنوب أفريقيا",
        "displayName-count-one";
        "راند جنوب أفريقيا",
        "displayName-count-two";
        "راند جنوب أفريقيا",
        "displayName-count-few";
        "راند جنوب أفريقيا",
        "displayName-count-many";
        "راند جنوب أفريقيا",
        "displayName-count-other";
        "راند جنوب أفريقيا",
        "symbol";
        "ZAR",

```

```

        "symbol-alt-narrow";
        "R";
    }
    "ZMK";
    {
        "displayName";
        "2012-1968 - ڪواشا زامبي",
        "displayName-count-zero";
        "2012-1968 - ڪواشا زامبي",
        "displayName-count-one";
        "2012-1968 - ڪواشا زامبي",
        "displayName-count-two";
        "2012-1968 - ڪواشا زامبي",
        "displayName-count-few";
        "2012-1968 - ڪواشا زامبي",
        "displayName-count-many";
        "2012-1968 - ڪواشا زامبي",
        "displayName-count-other";
        "2012-1968 - ڪواشا زامبي",
        "symbol";
        "ZMK";
    }
    "ZMW";
    {
        "displayName";
        "ڪواشا زامبي",
        "displayName-count-zero";
        "ڪواشا زامبي",
        "displayName-count-one";
        "ڪواشا زامبي",
        "displayName-count-two";
        "ڪواشا زامبي",
        "displayName-count-few";
        "ڪواشا زامبي",
        "displayName-count-many";
        "ڪواشا زامبي",
        "displayName-count-other";
        "ڪواشا زامبي",
        "symbol";
        "ZMW",
        "symbol-alt-narrow";
        "ZK";
    }
    "ZRN";
    {
        "displayName";
        "زائير زائيري جديد",
        "symbol";
        "ZRN";
    }
    "ZRZ";
    {
        "displayName";
        "زائير زائيري",
        "symbol";
        "ZRZ";
    }
}

```

```

        "ZWD";
    {
        "displayName";
        "دولار زمبابوي",
        "symbol";
        "ZWD";
    }
    "ZWL";
    {
        "displayName";
        "2009 دولار زمبابوي",
        "symbol";
        "ZWL";
    }
    "ZWR";
    {
        "displayName";
        "ZWR",
        "symbol";
        "ZWR";
    }
}
}
}
}
}

```

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
//import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import { loadCldr, L10n } from '@syncfusion/ej2-base';
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as timeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames);
L10n.load({
  'ar': {
    'calendar': {
      today: "اليوم"
    }
  }
});
function App() {
  return <CalendarComponent id="calendar" locale='ar' enableRtl={true}/>;
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import * as React from "react";
```

```
import * as ReactDOM from "react-dom";
//import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import { loadCldr, L10n } from '@syncfusion/ej2-base';
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as timeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames);
L10n.load({
    'ar': {
        'calendar': {
            today: "اليوم"
        }
    }
});
function App() {
    return <CalendarComponent id="calendar" locale='ar' enableRtl={true} />
};
ReactDOM.render(<App />, document.getElementById('element'));
```

NUMBERINGSYSTEMS.JSON

```
{
  "supplemental": {
    "version": {
      "_number": "$Revision: 12732 $",
      "_unicodeVersion": "9.0.0",
      "_cldrVersion": "31"
    },
    "numberingSystems": {
      "adlm": {
        "_digits": "ᲠᲡᲢᲣᲤᲥᲦᲧᲨᲩ",
        "_type": "numeric"
      },
      "ahom": {
        "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊",
        "_type": "numeric"
      },
      "arab": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "arabext": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "armn": {
        "_rules": "armenian-upper",
        "_type": "algorithmic"
      },
      "armnlow": {
        "_rules": "armenian-lower",
        "_type": "algorithmic"
      }
    }
  },
}
```



```
"bali": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"beng": {
  "_digits": "০১২৩৪৫৬৭৮৯",
  "_type": "numeric"
},
"bhks": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"brah": {
  "_digits": "·ᱠᱡᱢᱫᱷᱚᱨᱥᱤ",
  "_type": "numeric"
},
"cakm": {
  "_digits": "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
  "_type": "numeric"
},
"cham": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"cyrl": {
  "_rules": "cyrillic-lower",
  "_type": "algorithmic"
},
"deva": {
  "_digits": "०१२३४५६७८९",
  "_type": "numeric"
},
"ethi": {
  "_rules": "ethiopic",
  "_type": "algorithmic"
},
"fullwide": {
  "_digits": "0 1 2 3 4 5 6 7 8 9",
  "_type": "numeric"
},
"geor": {
  "_rules": "georgian",
  "_type": "algorithmic"
},
"grek": {
  "_rules": "greek-upper",
  "_type": "algorithmic"
},
"greklow": {
  "_rules": "greek-lower",
  "_type": "algorithmic"
},
"gujr": {
  "_digits": "૦૧૨૩૪૫૬૭૮૯",
  "_type": "numeric"
}
```

```

    },
    "guru": {
      "_digits": "༠༡༢༣༤༥༦༧༨༩",
      "_type": "numeric"
    },
    "hanidays": {
      "_rules": "zh/SpelloutRules/spellout-numbering-days",
      "_type": "algorithmic"
    },
    "hanidec": {
      "_digits": "〇一二三四五六七八九",
      "_type": "numeric"
    },
    "hans": {
      "_rules": "zh/SpelloutRules/spellout-cardinal",
      "_type": "algorithmic"
    },
    "hansfin": {
      "_rules": "zh/SpelloutRules/spellout-cardinal-financial",
      "_type": "algorithmic"
    },
    "hant": {
      "_rules": "zh_Hant/SpelloutRules/spellout-cardinal",
      "_type": "algorithmic"
    },
    "hantfin": {
      "_rules": "zh_Hant/SpelloutRules/spellout-cardinal-financial",
      "_type": "algorithmic"
    },
    "hebr": {
      "_rules": "hebrew",
      "_type": "algorithmic"
    },
    "hmng": {
      "_digits": "᠐᠑᠒᠐ᠠᠨᠠᠭ",
      "_type": "numeric"
    },
    "java": {
      "_digits": "᠐ᠠᠨᠠᠭᠤᠯᠤᠰᠤᠨᠠᠭᠤᠯᠤᠰ",
      "_type": "numeric"
    },
    "jpan": {
      "_rules": "ja/SpelloutRules/spellout-cardinal",
      "_type": "algorithmic"
    },
    "jpanfin": {
      "_rules": "ja/SpelloutRules/spellout-cardinal-financial",
      "_type": "algorithmic"
    },
    "kali": {
      "_digits": "᠐ᠠᠨᠠᠭᠤᠯᠤᠰᠤᠨᠠᠭᠤᠯᠤᠰ",
      "_type": "numeric"
    },
    "khmr": {
      "_digits": "០១២៣៤៥៦៧៨៩",

```

```

    "_type": "numeric"
  },
  "knda": {
    "_digits": "೦೧೨೩೪೫೬೭೮೯",
    "_type": "numeric"
  },
  "lana": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "lanatham": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "laoo": {
    "_digits": "໐໑໒໓໔໕໖໗໘໑",
    "_type": "numeric"
  },
  "latn": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "lepc": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "limb": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "mathbold": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mathdbl": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mathmono": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mathsanb": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mathsans": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mlym": {
    "_digits": "൦൧൨൩൪൫൬൭൮൯",
    "_type": "numeric"
  },
  "modi": {

```

```

    "_digits": "□□□□□□□□",
    "_type": "numeric"
  },
  "mong": {
    "_digits": "᠐᠑᠒᠓᠔᠕᠖᠗᠘᠐",
    "_type": "numeric"
  },
  "mroo": {
    "_digits": "□□□□□□□□",
    "_type": "numeric"
  },
  "mtei": {
    "_digits": "᠐᠑ᠶ᠋ᠸᠹᠺᠻᠼᠽ",
    "_type": "numeric"
  },
  "mymr": {
    "_digits": "၀၁၂၃၄၅၆၇၈",
    "_type": "numeric"
  },
  "mymrshan": {
    "_digits": "012345678",
    "_type": "numeric"
  },
  "mymrtlng": {
    "_digits": "᠐ᠠᠨᠵᠢᠨᠠᠨᠢᠨᠠ",
    "_type": "numeric"
  },
  "newa": {
    "_digits": "□□□□□□□□",
    "_type": "numeric"
  },
  "nkoo": {
    "_digits": "᠐ᠠᠨᠵᠢᠨᠠᠨᠢᠨᠠ",
    "_type": "numeric"
  },
  "olck": {
    "_digits": "᠐ᠠᠨᠵᠢᠨᠠᠨᠢᠨᠠ",
    "_type": "numeric"
  },
  "orya": {
    "_digits": "୦୧୨୩୪୫୬୭୮୯",
    "_type": "numeric"
  },
  "osma": {
    "_digits": "᠐᠑ᠶ᠋ᠸᠹᠺᠻᠼᠽ",
    "_type": "numeric"
  },
  "roman": {
    "_rules": "roman-upper",
    "_type": "algorithmic"
  },
  "romanlow": {
    "_rules": "roman-lower",

```

```

    "_type": "algorithmic"
  },
  "saur": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "shrd": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "sind": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "sinh": {
    "_digits": "ආශ්වතෙරභාදරාසච්ඡෙදරභා",
    "_type": "numeric"
  },
  "sora": {
    "_digits": "0෦෦෦෦෦෦෦෦෦෦",
    "_type": "numeric"
  },
  "sund": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "takr": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "talv": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "taml": {
    "_rules": "tamil",
    "_type": "algorithmic"
  },
  "tamldec": {
    "_digits": "0௧௨௩௪௫௬௭௮௯",
    "_type": "numeric"
  },
  "telu": {
    "_digits": "0౧౨౩౪౫౬౭౮౯",
    "_type": "numeric"
  },
  "thai": {
    "_digits": "๐๑๒๓๔๕๖๗๘๙",
    "_type": "numeric"
  },
  "tibv": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "tirh": {

```



```

        "_type";
        "numeric";
    }
    "armn";
    {
        "_rules";
        "armenian-upper",
        "_type";
        "algorithmic";
    }
    "armnlow";
    {
        "_rules";
        "armenian-lower",
        "_type";
        "algorithmic";
    }
    "bali";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "beng";
    {
        "_digits";
        "০১২৩৪৫৬৭৮৯",
        "_type";
        "numeric";
    }
    "bhks";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "brah";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "cakm";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "cham";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",

```

```

        "_type";
        "numeric";
    }
    "cyrl";
    {
        "_rules";
        "cyrillic-lower",
        "_type";
        "algorithmic";
    }
    "deva";
    {
        "_digits";
        "०१२३४५६७८९",
        "_type";
        "numeric";
    }
    "ethi";
    {
        "_rules";
        "ethiopic",
        "_type";
        "algorithmic";
    }
    "fullwide";
    {
        "_digits";
        "0 1 2 3 4 5 6 7 8 9",
        "_type";
        "numeric";
    }
    "geor";
    {
        "_rules";
        "georgian",
        "_type";
        "algorithmic";
    }
    "grek";
    {
        "_rules";
        "greek-upper",
        "_type";
        "algorithmic";
    }
    "greklow";
    {
        "_rules";
        "greek-lower",
        "_type";
        "algorithmic";
    }
    "gujr";
    {
        "_digits";
        "૦૧૨૩૪૫૬૭૮૯",

```



```

        "_type";
        "numeric";
    }
    "guru";
    {
        "_digits";
        "୦୧୨୩୪୫୬୭୮୯",
        "_type";
        "numeric";
    }
    "hanidays";
    {
        "_rules";
        "zh/SpelloutRules/spellout-numbering-days",
        "_type";
        "algorithmic";
    }
    "hanidec";
    {
        "_digits";
        "〇一二三四五六七八九",
        "_type";
        "numeric";
    }
    "hans";
    {
        "_rules";
        "zh/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "hansfin";
    {
        "_rules";
        "zh/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "hant";
    {
        "_rules";
        "zh_Hant/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "hantfin";
    {
        "_rules";
        "zh_Hant/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "hebr";
    {
        "_rules";
        "hebrew",

```

```

        "_type";
        "algorithmic";
    }
    "hmng";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "java";
    {
        "_digits";
        "၀၈၈၅၅၆၆၈၈၈၈၈",
        "_type";
        "numeric";
    }
    "jpan";
    {
        "_rules";
        "ja/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "jpanfin";
    {
        "_rules";
        "ja/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "kali";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "khdr";
    {
        "_digits";
        "၀၅၅၈၈၈၈၈၈၈၈",
        "_type";
        "numeric";
    }
    "knda";
    {
        "_digits";
        "೦೧೨೩೪೫೬೭೮೯",
        "_type";
        "numeric";
    }
    "lana";
    {
        "_digits";

```

```

        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "lanatham";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "laoo";
    {
        "_digits";
        "໐໑໒໓໔໕໖໗໘໑",
        "_type";
        "numeric";
    }
    "latn";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "lepc";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "limb";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "mathbold";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathdbl";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathmono";
    {
        "_digits";

```

```

        "0123456789",
        "_type";
        "numeric";
    }
    "mathsanb";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathsans";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mlym";
    {
        "_digits";
        "ഐറുനൂറുനൂറു",
        "_type";
        "numeric";
    }
    "modi";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "mong";
    {
        "_digits";
        "᠎ᠠᠨᠣᠭᠤ",
        "_type";
        "numeric";
    }
    "mroo";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "mtei";
    {
        "_digits";
        "ၵၵၵၵၵၵၵၵ",
        "_type";
        "numeric";
    }
    "mymr";
    {
        "_digits";

```

```

        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "mymrshan";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "mymrtlng";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "newa";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "nkoo";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "olck";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "orya";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "osma";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "roman";

```

```

    {
      "_rules";
      "roman-upper",
      "_type";
      "algorithmic";
    }
    "romanlow";
    {
      "_rules";
      "roman-lower",
      "_type";
      "algorithmic";
    }
    "saur";
    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "shrd";
    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "sind";
    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "sinh";
    {
      "_digits";
      "𑆑𑆒𑆓𑆔𑆕𑆖𑆗𑆘",
      "_type";
      "numeric";
    }
    "sora";
    {
      "_digits";
      "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉",
      "_type";
      "numeric";
    }
    "sund";
    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "takr";

```

```

    {
      "_digits";
      "□□□□□□□□□□",
      "_type";
      "numeric";
    }
    "tal";
    {
      "_digits";
      "௧௨௩௪௫௬௭௮௯௦",
      "_type";
      "numeric";
    }
    "tam";
    {
      "_rules";
      "tamil",
      "_type";
      "algorithmic";
    }
    "tamld";
    {
      "_digits";
      "௦௧௨௩௪௫௬௭௮௯௦",
      "_type";
      "numeric";
    }
    "tel";
    {
      "_digits";
      "౦౧౨౩౪౫౬౭౮౯",
      "_type";
      "numeric";
    }
    "thai";
    {
      "_digits";
      "๐๑๒๓๔๕๖๗๘๙",
      "_type";
      "numeric";
    }
    "tib";
    {
      "_digits";
      "༠༡༢༣༤༥༦༧༨༩",
      "_type";
      "numeric";
    }
    "tir";
    {
      "_digits";
      "□□□□□□□□□□",
      "_type";
      "numeric";
    }
    "vai";

```

NUMBERS.JSON

Copyright © 2001 -2024 Syncfusion Inc.


```

    "exponential": "E",
    "superscriptingExponent": "×",
    "perMille": "%",
    "infinity": "∞",
    "nan": "ليس رقمًا",
    "timeSeparator": ":"
  },
  "decimalFormats-numberSystem-arab": {
    "standard": "#,##0.###",
    "long": {
      "decimalFormat": {
        "1000-count-zero": "0 ألف",
        "1000-count-one": "0 ألف",
        "1000-count-two": "0 ألف",
        "1000-count-few": "0 آلاف",
        "1000-count-many": "0 ألف",
        "1000-count-other": "0 ألف",
        "10000-count-zero": "00 ألف",
        "10000-count-one": "00 ألف",
        "10000-count-two": "00 ألف",
        "10000-count-few": "00 ألف",
        "10000-count-many": "00 ألف",
        "10000-count-other": "00 ألف",
        "100000-count-zero": "000 ألف",
        "100000-count-one": "000 ألف",
        "100000-count-two": "000 ألف",
        "100000-count-few": "000 ألف",
        "100000-count-many": "000 ألف",
        "100000-count-other": "000 ألف",
        "1000000-count-zero": "0 مليون",
        "1000000-count-one": "0 مليون",
        "1000000-count-two": "0 مليون",
        "1000000-count-few": "0 ملايين",
        "1000000-count-many": "0 مليون",
        "1000000-count-other": "0 مليون",
        "10000000-count-zero": "00 مليون",
        "10000000-count-one": "00 مليون",
        "10000000-count-two": "00 مليون",
        "10000000-count-few": "00 ملايين",
        "10000000-count-many": "00 مليون",
        "10000000-count-other": "00 مليون",
        "100000000-count-zero": "000 مليون",
        "100000000-count-one": "000 مليون",
        "100000000-count-two": "000 مليون",
        "100000000-count-few": "000 مليون",
        "100000000-count-many": "000 مليون",
        "100000000-count-other": "000 مليون",
        "1000000000-count-zero": "0 مليار",
        "1000000000-count-one": "0 مليار",
        "1000000000-count-two": "0 مليار",
        "1000000000-count-few": "0 مليار",
        "1000000000-count-many": "0 مليار",
        "1000000000-count-other": "0 مليار",
        "10000000000-count-zero": "00 مليار",
        "10000000000-count-one": "00 مليار",
        "10000000000-count-two": "00 مليار",
        "10000000000-count-few": "00 مليار",

```

```

        "10000000000-count-many": "00 مليار",
        "10000000000-count-other": "00 مليار",
        "10000000000-count-zero": "000 مليار",
        "10000000000-count-one": "000 مليار",
        "10000000000-count-two": "000 مليار",
        "10000000000-count-few": "000 مليار",
        "10000000000-count-many": "000 مليار",
        "10000000000-count-other": "000 مليار",
        "10000000000-count-zero": "0 تريليون",
        "10000000000-count-one": "0 تريليون",
        "10000000000-count-two": "0 تريليون",
        "10000000000-count-few": "0 تريليونات",
        "10000000000-count-many": "0 تريليون",
        "10000000000-count-other": "0 تريليون",
        "10000000000-count-zero": "00 تريليون",
        "10000000000-count-one": "00 تريليون",
        "10000000000-count-two": "00 تريليون",
        "10000000000-count-few": "00 تريليون",
        "10000000000-count-many": "00 تريليون",
        "10000000000-count-other": "00 تريليون",
        "10000000000-count-zero": "000 تريليون",
        "10000000000-count-one": "000 تريليون",
        "10000000000-count-two": "000 تريليون",
        "10000000000-count-few": "000 تريليون",
        "10000000000-count-many": "000 تريليون",
        "10000000000-count-other": "000 تريليون"
    },
    },
    "short": {
        "decimalFormat": {
            "1000-count-zero": "0 ألف",
            "1000-count-one": "0 ألف",
            "1000-count-two": "0 ألف",
            "1000-count-few": "0 آلاف",
            "1000-count-many": "0 ألف",
            "1000-count-other": "0 ألف",
            "10000-count-zero": "00 ألف",
            "10000-count-one": "00 ألف",
            "10000-count-two": "00 ألف",
            "10000-count-few": "00 ألف",
            "10000-count-many": "00 ألف",
            "10000-count-other": "00 ألف",
            "100000-count-zero": "000 ألف",
            "100000-count-one": "000 ألف",
            "100000-count-two": "000 ألف",
            "100000-count-few": "000 ألف",
            "100000-count-many": "000 ألف",
            "100000-count-other": "000 ألف",
            "1000000-count-zero": "0 مليون",
            "1000000-count-one": "0 مليون",
            "1000000-count-two": "0 مليون",
            "1000000-count-few": "0 مليون",
            "1000000-count-many": "0 مليون",
            "1000000-count-other": "0 مليون",
            "10000000-count-zero": "00 مليون",
            "10000000-count-one": "00 مليون",
            "10000000-count-two": "00 مليون",

```

```

        "10000000-count-few": "00 مليو",
        "10000000-count-many": "00 مليو",
        "10000000-count-other": "00 مليو",
        "10000000-count-zero": "000 مليو",
        "100000000-count-one": "000 مليو",
        "100000000-count-two": "000 مليو",
        "100000000-count-few": "000 مليو",
        "100000000-count-many": "000 مليو",
        "100000000-count-other": "000 مليو",
        "1000000000-count-zero": "0 مليا",
        "1000000000-count-one": "0 مليا",
        "1000000000-count-two": "0 مليا",
        "1000000000-count-few": "0 مليا",
        "1000000000-count-many": "0 مليا",
        "1000000000-count-other": "0 مليا",
        "10000000000-count-zero": "00 مليا",
        "10000000000-count-one": "00 مليا",
        "10000000000-count-two": "00 مليا",
        "10000000000-count-few": "00 مليا",
        "10000000000-count-many": "00 مليا",
        "10000000000-count-other": "00 مليا",
        "100000000000-count-zero": "000 مليا",
        "100000000000-count-one": "000 مليا",
        "100000000000-count-two": "000 مليا",
        "100000000000-count-few": "000 مليا",
        "100000000000-count-many": "000 مليا",
        "100000000000-count-other": "000 مليا",
        "1000000000000-count-zero": "0 ترليو",
        "1000000000000-count-one": "0 ترليو",
        "1000000000000-count-two": "0 ترليو",
        "1000000000000-count-few": "0 ترليو",
        "1000000000000-count-many": "0 ترليو",
        "1000000000000-count-other": "0 ترليو",
        "10000000000000-count-zero": "00 ترليو",
        "10000000000000-count-one": "00 ترليو",
        "10000000000000-count-two": "00 ترليو",
        "10000000000000-count-few": "00 ترليو",
        "10000000000000-count-many": "00 ترليو",
        "10000000000000-count-other": "00 ترليو",
        "100000000000000-count-zero": "000 ترليو",
        "100000000000000-count-one": "000 ترليو",
        "100000000000000-count-two": "000 ترليو",
        "100000000000000-count-few": "000 ترليو",
        "100000000000000-count-many": "000 ترليو",
        "100000000000000-count-other": "000 ترليو"
    }
}
},
"decimalFormats-numberSystem-latn": {
    "standard": "#,##0.###",
    "long": {
        "decimalFormat": {
            "1000-count-zero": "0 ألف",
            "1000-count-one": "0 ألف",
            "1000-count-two": "0 ألف",
            "1000-count-few": "0 آلاف",
            "1000-count-many": "0 ألف",
        }
    }
}

```

```
"1000-count-other": "0 ألف",
"10000-count-zero": "00 ألف",
"10000-count-one": "00 ألف",
"10000-count-two": "00 ألف",
"10000-count-few": "00 ألف",
"10000-count-many": "00 ألف",
"10000-count-other": "00 ألف",
"100000-count-zero": "000 ألف",
"100000-count-one": "000 ألف",
"100000-count-two": "000 ألف",
"100000-count-few": "000 ألف",
"100000-count-many": "000 ألف",
"100000-count-other": "000 ألف",
"1000000-count-zero": "0 مليون",
"1000000-count-one": "0 مليون",
"1000000-count-two": "0 مليون",
"1000000-count-few": "0 ملايين",
"1000000-count-many": "0 مليون",
"1000000-count-other": "0 مليون",
"10000000-count-zero": "00 مليون",
"10000000-count-one": "00 مليون",
"10000000-count-two": "00 مليون",
"10000000-count-few": "00 ملايين",
"10000000-count-many": "00 مليون",
"10000000-count-other": "00 مليون",
"100000000-count-zero": "000 مليون",
"100000000-count-one": "000 مليون",
"100000000-count-two": "000 مليون",
"100000000-count-few": "000 مليون",
"100000000-count-many": "000 مليون",
"100000000-count-other": "000 مليون",
"1000000000-count-zero": "0 مليار",
"1000000000-count-one": "0 مليار",
"1000000000-count-two": "0 مليار",
"1000000000-count-few": "0 مليار",
"1000000000-count-many": "0 مليار",
"1000000000-count-other": "0 مليار",
"10000000000-count-zero": "00 مليار",
"10000000000-count-one": "00 مليار",
"10000000000-count-two": "00 مليار",
"10000000000-count-few": "00 مليار",
"10000000000-count-many": "00 مليار",
"10000000000-count-other": "00 مليار",
"100000000000-count-zero": "0 تريليون",
"100000000000-count-one": "0 تريليون",
"100000000000-count-two": "0 تريليون",
"100000000000-count-few": "0 تريليونات",
"100000000000-count-many": "0 تريليون",
"100000000000-count-other": "0 تريليون",
"1000000000000-count-zero": "00 تريليون",
"1000000000000-count-one": "00 تريليون",
```

```

        "10000000000000-count-two": "00 تريليون",
        "10000000000000-count-few": "00 تريليون",
        "10000000000000-count-many": "00 تريليون",
        "10000000000000-count-other": "00 تريليون",
        "10000000000000-count-zero": "000 تريليون",
        "10000000000000-count-one": "000 تريليون",
        "10000000000000-count-two": "000 تريليون",
        "10000000000000-count-few": "000 تريليون",
        "10000000000000-count-many": "000 تريليون",
        "10000000000000-count-other": "000 تريليون"
    },
    },
    "short": {
        "decimalFormat": {
            "1000-count-zero": "0 ألف",
            "1000-count-one": "0 ألف",
            "1000-count-two": "0 ألف",
            "1000-count-few": "0 آلاف",
            "1000-count-many": "0 ألف",
            "1000-count-other": "0 ألف",
            "10000-count-zero": "00 ألف",
            "10000-count-one": "00 ألف",
            "10000-count-two": "00 ألف",
            "10000-count-few": "00 ألف",
            "10000-count-many": "00 ألف",
            "10000-count-other": "00 ألف",
            "100000-count-zero": "000 ألف",
            "100000-count-one": "000 ألف",
            "100000-count-two": "000 ألف",
            "100000-count-few": "000 ألف",
            "100000-count-many": "000 ألف",
            "100000-count-other": "000 ألف",
            "1000000-count-zero": "0 مليو",
            "1000000-count-one": "0 مليو",
            "1000000-count-two": "0 مليو",
            "1000000-count-few": "0 مليو",
            "1000000-count-many": "0 مليو",
            "1000000-count-other": "0 مليو",
            "10000000-count-zero": "00 مليو",
            "10000000-count-one": "00 مليو",
            "10000000-count-two": "00 مليو",
            "10000000-count-few": "00 مليو",
            "10000000-count-many": "00 مليو",
            "10000000-count-other": "00 مليو",
            "100000000-count-zero": "000 مليو",
            "100000000-count-one": "000 مليو",
            "100000000-count-two": "000 مليو",
            "100000000-count-few": "000 مليو",
            "100000000-count-many": "000 مليو",
            "100000000-count-other": "000 مليو",
            "1000000000-count-zero": "0 مليا",
            "1000000000-count-one": "0 مليا",
            "1000000000-count-two": "0 مليا",
            "1000000000-count-few": "0 مليا",
            "1000000000-count-many": "0 مليا",
            "1000000000-count-other": "0 مليا",
            "10000000000-count-zero": "00 مليا",

```

```

        "10000000000-count-one": "00 مليا",
        "10000000000-count-two": "00 مليا",
        "10000000000-count-few": "00 مليا",
        "10000000000-count-many": "00 مليا",
        "10000000000-count-other": "00 مليا",
        "10000000000-count-zero": "000 مليا",
        "100000000000-count-one": "000 مليا",
        "100000000000-count-two": "000 مليا",
        "100000000000-count-few": "000 مليا",
        "100000000000-count-many": "000 مليا",
        "100000000000-count-other": "000 مليا",
        "1000000000000-count-zero": "0 ترليو",
        "1000000000000-count-one": "0 ترليو",
        "1000000000000-count-two": "0 ترليو",
        "1000000000000-count-few": "0 ترليو",
        "1000000000000-count-many": "0 ترليو",
        "1000000000000-count-other": "0 ترليو",
        "10000000000000-count-zero": "00 ترليو",
        "10000000000000-count-one": "00 ترليو",
        "10000000000000-count-two": "00 ترليو",
        "10000000000000-count-few": "00 ترليو",
        "10000000000000-count-many": "00 ترليو",
        "10000000000000-count-other": "00 ترليو",
        "100000000000000-count-zero": "000 ترليو",
        "100000000000000-count-one": "000 ترليو",
        "100000000000000-count-two": "000 ترليو",
        "100000000000000-count-few": "000 ترليو",
        "100000000000000-count-many": "000 ترليو",
        "100000000000000-count-other": "000 ترليو"
    }
}
},
"scientificFormats-numberSystem-arab": {
    "standard": "#E0"
},
"scientificFormats-numberSystem-latn": {
    "standard": "#E0"
},
"percentFormats-numberSystem-arab": {
    "standard": "#,##0 %"
},
"percentFormats-numberSystem-latn": {
    "standard": "#,##0%"
},
"currencyFormats-numberSystem-arab": {
    "currencySpacing": {
        "beforeCurrency": {
            "currencyMatch": "[:^S:]",
            "surroundingMatch": "[:digit:]",
            "insertBetween": " "
        },
        "afterCurrency": {
            "currencyMatch": "[:^S:]",
            "surroundingMatch": "[:digit:]",
            "insertBetween": " "
        }
    }
},
},

```

```

    "standard": "#,##0.00 ¤",
    "accounting": "#,##0.00 ¤",
    "unitPattern-count-zero": "{0} {1}",
    "unitPattern-count-one": "{0} {1}",
    "unitPattern-count-two": "{0} {1}",
    "unitPattern-count-few": "{0} {1}",
    "unitPattern-count-many": "{0} {1}",
    "unitPattern-count-other": "{0} {1}"
  },
  "currencyFormats-numberSystem-latn": {
    "currencySpacing": {
      "beforeCurrency": {
        "currencyMatch": "[:^S:]",
        "surroundingMatch": "[:digit:]",
        "insertBetween": " "
      },
      "afterCurrency": {
        "currencyMatch": "[:^S:]",
        "surroundingMatch": "[:digit:]",
        "insertBetween": " "
      }
    },
    "standard": "¤ #,##0.00",
    "accounting": "¤#,##0.00; (¤#,##0.00)",
    "short": {
      "standard": {
        "1000-count-zero": "¤ 0 ألف",
        "1000-count-one": "¤ 0 ألف",
        "1000-count-two": "¤ 0 ألف",
        "1000-count-few": "¤ 0 ألف",
        "1000-count-many": "¤ 0 ألف",
        "1000-count-other": "¤ 0 ألف",
        "10000-count-zero": "¤ 00 ألف",
        "10000-count-one": "¤ 00 ألف",
        "10000-count-two": "¤ 00 ألف",
        "10000-count-few": "¤ 00 ألف",
        "10000-count-many": "¤ 00 ألف",
        "10000-count-other": "¤ 00 ألف",
        "100000-count-zero": "¤ 000 ألف",
        "100000-count-one": "¤ 000 ألف",
        "100000-count-two": "¤ 000 ألف",
        "100000-count-few": "¤ 000 ألف",
        "100000-count-many": "¤ 000 ألف",
        "100000-count-other": "¤ 000 ألف",
        "1000000-count-zero": "¤ 0 مليون",
        "1000000-count-one": "¤ 0 مليون",
        "1000000-count-two": "¤ 0 مليون",
        "1000000-count-few": "¤ 0 مليون",
        "1000000-count-many": "¤ 0 مليون",
        "1000000-count-other": "¤ 0 مليون",
        "10000000-count-zero": "¤ 00 مليون",
        "10000000-count-one": "¤ 00 مليون",
        "10000000-count-two": "¤ 00 مليون",
        "10000000-count-few": "¤ 00 مليون",
        "10000000-count-many": "¤ 00 مليون",
        "10000000-count-other": "¤ 00 مليون",
        "100000000-count-zero": "¤ 000 مليون"
      }
    }
  }
}

```

```

        "1000000000-count-one": "١٠٠٠ ٠٠٠ مليو",
        "1000000000-count-two": "١٠٠٠ ٠٠٠ مليو",
        "1000000000-count-few": "١٠٠٠ ٠٠٠ مليو",
        "1000000000-count-many": "١٠٠٠ ٠٠٠ مليو",
        "1000000000-count-other": "١٠٠٠ ٠٠٠ مليو",
        "1000000000-count-zero": "١٠٠٠ ٠ مليا",
        "1000000000-count-one": "١٠٠٠ ٠ مليا",
        "1000000000-count-two": "١٠٠٠ ٠ مليا",
        "1000000000-count-few": "١٠٠٠ ٠ مليا",
        "1000000000-count-many": "١٠٠٠ ٠ مليا",
        "1000000000-count-other": "١٠٠٠ ٠ مليا",
        "1000000000-count-zero": "١٠٠٠ ٠٠ مليا",
        "10000000000-count-one": "١٠٠٠٠ ٠٠ مليا",
        "10000000000-count-two": "١٠٠٠٠ ٠٠ مليا",
        "10000000000-count-few": "١٠٠٠٠ ٠٠ مليا",
        "10000000000-count-many": "١٠٠٠٠ ٠٠ مليا",
        "10000000000-count-other": "١٠٠٠٠ ٠٠ مليا",
        "100000000000-count-zero": "١٠٠٠٠٠ ٠٠٠ مليا",
        "100000000000-count-one": "١٠٠٠٠٠ ٠٠٠ مليا",
        "100000000000-count-two": "١٠٠٠٠٠ ٠٠٠ مليا",
        "100000000000-count-few": "١٠٠٠٠٠ ٠٠٠ مليا",
        "100000000000-count-many": "١٠٠٠٠٠ ٠٠٠ مليا",
        "100000000000-count-other": "١٠٠٠٠٠ ٠٠٠ مليا",
        "1000000000000-count-zero": "١٠٠٠٠٠٠ ٠ ترليو",
        "1000000000000-count-one": "١٠٠٠٠٠٠ ٠ ترليو",
        "1000000000000-count-two": "١٠٠٠٠٠٠ ٠ ترليو",
        "1000000000000-count-few": "١٠٠٠٠٠٠ ٠ ترليو",
        "1000000000000-count-many": "١٠٠٠٠٠٠ ٠ ترليو",
        "1000000000000-count-other": "١٠٠٠٠٠٠ ٠ ترليو",
        "10000000000000-count-zero": "١٠٠٠٠٠٠٠ ٠٠ ترليو",
        "10000000000000-count-one": "١٠٠٠٠٠٠٠ ٠٠ ترليو",
        "10000000000000-count-two": "١٠٠٠٠٠٠٠ ٠٠ ترليو",
        "10000000000000-count-few": "١٠٠٠٠٠٠٠ ٠٠ ترليو",
        "10000000000000-count-many": "١٠٠٠٠٠٠٠ ٠٠ ترليو",
        "10000000000000-count-other": "١٠٠٠٠٠٠٠ ٠٠ ترليو",
        "100000000000000-count-zero": "١٠٠٠٠٠٠٠٠ ٠٠٠ ترليو",
        "100000000000000-count-one": "١٠٠٠٠٠٠٠٠ ٠٠٠ ترليو",
        "100000000000000-count-two": "١٠٠٠٠٠٠٠٠ ٠٠٠ ترليو",
        "100000000000000-count-few": "١٠٠٠٠٠٠٠٠ ٠٠٠ ترليو",
        "100000000000000-count-many": "١٠٠٠٠٠٠٠٠ ٠٠٠ ترليو",
        "100000000000000-count-other": "١٠٠٠٠٠٠٠٠ ٠٠٠ ترليو"
    },
    },
    "unitPattern-count-zero": "{0} {1}",
    "unitPattern-count-one": "{0} {1}",
    "unitPattern-count-two": "{0} {1}",
    "unitPattern-count-few": "{0} {1}",
    "unitPattern-count-many": "{0} {1}",
    "unitPattern-count-other": "{0} {1}"
},
"miscPatterns-numberSystem-arab": {
    "atLeast": "+{0}",
    "range": "{0}-{1}"
},
"miscPatterns-numberSystem-latn": {
    "atLeast": "+{0}",
    "range": "{0}-{1}"
}

```



```

    }
  }
}
}
}

```

NUMBERS.JSX

```

{
  "main";
  {
    "ar";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 12879 $",
          "_cldrVersion";
          "30.0.3";
        }
        "language";
        "ar";
      }
    }
    "numbers";
    {
      "defaultNumberingSystem";
      "arab",
      "otherNumberingSystems";
      {
        "native";
        "arab";
      }
      "minimumGroupingDigits";
      "1",
      "symbols-numberSystem-arab";
      {
        "decimal";
        ",",
        "group";
        "list";
        ":",
        "percentSign";
        "%",
        "plusSign";
        "+",
        "minusSign";
        "-",
        "exponential";
        "٭",
        "superscriptingExponent";
        "×",
        "perMille";
        "‰",

```

```

        "infinity";
        "∞",
        "nan";
        "ليس رقم",
        "timeSeparator";
        ":";
    }
    "symbols-numberSystem-latn";
    {
        "decimal";
        ".",
        "group";
        ",",
        "list";
        ";",
        "percentSign";
        "%",
        "plusSign";
        "+",
        "minusSign";
        "-",
        "exponential";
        "E",
        "superscriptingExponent";
        "×",
        "perMille";
        "‰",
        "infinity";
        "∞",
        "nan";
        "ليس رقمًا",
        "timeSeparator";
        ":";
    }
    "decimalFormats-numberSystem-arab";
    {
        "standard";
        "#,##0.###",
        "long";
        {
            "decimalFormat";
            {
                "1000-count-zero";
                "0 ألف",
                "1000-count-one";
                "0 ألف",
                "1000-count-two";
                "0 ألف",
                "1000-count-few";
                "0 آلاف",
                "1000-count-many";
                "0 ألف",
                "1000-count-other";
                "0 ألف",
                "10000-count-zero";
                "00 ألف",
                "10000-count-one";
            }
        }
    }

```

```

"00 ألف",
  "10000-count-two";
"00 ألف",
  "10000-count-few";
"00 ألف",
  "10000-count-many";
"00 ألف",
  "10000-count-other";
"00 ألف",
  "100000-count-zero";
"000 ألف",
  "100000-count-one";
"000 ألف",
  "100000-count-two";
"000 ألف",
  "100000-count-few";
"000 ألف",
  "100000-count-many";
"000 ألف",
  "100000-count-other";
"000 ألف",
  "1000000-count-zero";
"0 مليون",
  "1000000-count-one";
"0 مليون",
  "1000000-count-two";
"0 مليون",
  "1000000-count-few";
"0 ملايين",
  "1000000-count-many";
"0 مليون",
  "1000000-count-other";
"0 مليون",
  "10000000-count-zero";
"00 مليون",
  "10000000-count-one";
"00 مليون",
  "10000000-count-two";
"00 مليون",
  "10000000-count-few";
"00 ملايين",
  "10000000-count-many";
"00 مليون",
  "10000000-count-other";
"00 مليون",
  "100000000-count-zero";
"000 مليون",
  "100000000-count-one";
"000 مليون",
  "100000000-count-two";
"000 مليون",
  "100000000-count-few";
"000 مليون",
  "100000000-count-many";
"000 مليون",
  "100000000-count-other";
"000 مليون",

```

```

        "1000000000-count-zero";
    "0 مليار",
        "1000000000-count-one";
    "0 مليار",
        "1000000000-count-two";
    "0 مليار",
        "1000000000-count-few";
    "0 مليار",
        "1000000000-count-many";
    "0 مليار",
        "1000000000-count-other";
    "0 مليار",
        "10000000000-count-zero";
    "00 مليار",
        "10000000000-count-one";
    "00 مليار",
        "10000000000-count-two";
    "00 مليار",
        "10000000000-count-few";
    "00 مليار",
        "10000000000-count-many";
    "00 مليار",
        "10000000000-count-other";
    "00 مليار",
        "100000000000-count-zero";
    "000 مليار",
        "100000000000-count-one";
    "000 مليار",
        "100000000000-count-two";
    "000 مليار",
        "100000000000-count-few";
    "000 مليار",
        "100000000000-count-many";
    "000 مليار",
        "100000000000-count-other";
    "000 مليار",
        "1000000000000-count-zero";
    "0 تريليون",
        "1000000000000-count-one";
    "0 تريليون",
        "1000000000000-count-two";
    "0 تريليون",
        "1000000000000-count-few";
    "0 تريليونات",
        "1000000000000-count-many";
    "0 تريليون",
        "1000000000000-count-other";
    "0 تريليون",
        "10000000000000-count-zero";
    "00 تريليون",
        "10000000000000-count-one";
    "00 تريليون",
        "10000000000000-count-two";
    "00 تريليون",
        "10000000000000-count-few";
    "00 تريليون",
        "10000000000000-count-many";

```

```

        "00 تريليون",
        "10000000000000-count-other";
        "00 تريليون",
        "10000000000000-count-zero";
        "000 تريليون",
        "10000000000000-count-one";
        "000 تريليون",
        "10000000000000-count-two";
        "000 تريليون",
        "10000000000000-count-few";
        "000 تريليون",
        "10000000000000-count-many";
        "000 تريليون",
        "10000000000000-count-other";
        "000 تريليون";
    }
}
"short";
{
    "decimalFormat";
    {
        "1000-count-zero";
        "0 ألف",
        "1000-count-one";
        "0 ألف",
        "1000-count-two";
        "0 ألف",
        "1000-count-few";
        "0 آلاف",
        "1000-count-many";
        "0 ألف",
        "1000-count-other";
        "0 ألف",
        "10000-count-zero";
        "00 ألف",
        "10000-count-one";
        "00 ألف",
        "10000-count-two";
        "00 ألف",
        "10000-count-few";
        "00 ألف",
        "10000-count-many";
        "00 ألف",
        "10000-count-other";
        "00 ألف",
        "100000-count-zero";
        "000 ألف",
        "100000-count-one";
        "000 ألف",
        "100000-count-two";
        "000 ألف",
        "100000-count-few";
        "000 ألف",
        "100000-count-many";
        "000 ألف",
        "100000-count-other";
        "000 ألف",
    }
}

```

```
"1000000-count-zero";
"0 مليو",
"1000000-count-one";
"0 مليو",
"1000000-count-two";
"0 مليو",
"1000000-count-few";
"0 مليو",
"1000000-count-many";
"0 مليو",
"1000000-count-other";
"0 مليو",
"10000000-count-zero";
"00 مليو",
"10000000-count-one";
"00 مليو",
"10000000-count-two";
"00 مليو",
"10000000-count-few";
"00 مليو",
"10000000-count-many";
"00 مليو",
"10000000-count-other";
"00 مليو",
"100000000-count-zero";
"000 مليو",
"100000000-count-one";
"000 مليو",
"100000000-count-two";
"000 مليو",
"100000000-count-few";
"000 مليو",
"100000000-count-many";
"000 مليو",
"100000000-count-other";
"000 مليو",
"1000000000-count-zero";
"0 مليا",
"1000000000-count-one";
"0 مليا",
"1000000000-count-two";
"0 مليا",
"1000000000-count-few";
"0 مليا",
"1000000000-count-many";
"0 مليا",
"1000000000-count-other";
"0 مليا",
"10000000000-count-zero";
"00 مليا",
"10000000000-count-one";
"00 مليا",
"10000000000-count-two";
"00 مليا",
"10000000000-count-few";
"00 مليا",
"10000000000-count-many";
```

```

        "00 مليا",
        "10000000000-count-other";
        "00 مليا",
        "10000000000-count-zero";
        "000 مليا",
        "10000000000-count-one";
        "000 مليا",
        "10000000000-count-two";
        "000 مليا",
        "10000000000-count-few";
        "000 مليا",
        "10000000000-count-many";
        "000 مليا",
        "10000000000-count-other";
        "000 مليا",
        "10000000000-count-zero";
        "0 ترليو",
        "10000000000-count-one";
        "0 ترليو",
        "10000000000-count-two";
        "0 ترليو",
        "10000000000-count-few";
        "0 ترليو",
        "10000000000-count-many";
        "0 ترليو",
        "10000000000-count-other";
        "0 ترليو",
        "10000000000-count-zero";
        "00 ترليو",
        "10000000000-count-one";
        "00 ترليو",
        "10000000000-count-two";
        "00 ترليو",
        "10000000000-count-few";
        "00 ترليو",
        "10000000000-count-many";
        "00 ترليو",
        "10000000000-count-other";
        "00 ترليو",
        "10000000000-count-zero";
        "000 ترليو",
        "10000000000-count-one";
        "000 ترليو",
        "10000000000-count-two";
        "000 ترليو",
        "10000000000-count-few";
        "000 ترليو",
        "10000000000-count-many";
        "000 ترليو",
        "10000000000-count-other";
        "000 ترليو";
    }
}
}
"decimalFormats-numberSystem-latn";
{
    "standard";

```

```

"#,##0.###",
    "long";
{
    "decimalFormat";
    {
        "1000-count-zero";
        "0 ألف",
        "1000-count-one";
        "0 ألف",
        "1000-count-two";
        "0 ألف",
        "1000-count-few";
        "0 آلاف",
        "1000-count-many";
        "0 ألف",
        "1000-count-other";
        "0 ألف",
        "10000-count-zero";
        "00 ألف",
        "10000-count-one";
        "00 ألف",
        "10000-count-two";
        "00 ألف",
        "10000-count-few";
        "00 ألف",
        "10000-count-many";
        "00 ألف",
        "10000-count-other";
        "00 ألف",
        "100000-count-zero";
        "000 ألف",
        "100000-count-one";
        "000 ألف",
        "100000-count-two";
        "000 ألف",
        "100000-count-few";
        "000 ألف",
        "100000-count-many";
        "000 ألف",
        "100000-count-other";
        "000 ألف",
        "1000000-count-zero";
        "0 مليون",
        "1000000-count-one";
        "0 مليون",
        "1000000-count-two";
        "0 مليون",
        "1000000-count-few";
        "0 ملايين",
        "1000000-count-many";
        "0 مليون",
        "1000000-count-other";
        "0 مليون",
        "10000000-count-zero";
        "00 مليون",
        "10000000-count-one";
        "00 مليون",
    }
}

```



```

        "10000000-count-two";
    "00 مليون",
    "10000000-count-few";
    "00 ملايين",
    "10000000-count-many";
    "00 مليون",
    "10000000-count-other";
    "00 مليون",
    "100000000-count-zero";
    "000 مليون",
    "100000000-count-one";
    "000 مليون",
    "100000000-count-two";
    "000 مليون",
    "100000000-count-few";
    "000 مليون",
    "100000000-count-many";
    "000 مليون",
    "100000000-count-other";
    "000 مليون",
    "1000000000-count-zero";
    "0 مليار",
    "1000000000-count-one";
    "0 مليار",
    "1000000000-count-two";
    "0 مليار",
    "1000000000-count-few";
    "0 مليار",
    "1000000000-count-many";
    "0 مليار",
    "1000000000-count-other";
    "0 مليار",
    "10000000000-count-zero";
    "00 مليار",
    "10000000000-count-one";
    "00 مليار",
    "10000000000-count-two";
    "00 مليار",
    "10000000000-count-few";
    "00 مليار",
    "10000000000-count-many";
    "00 مليار",
    "10000000000-count-other";
    "00 مليار",
    "100000000000-count-zero";
    "000 مليار",
    "100000000000-count-one";
    "000 مليار",
    "100000000000-count-two";
    "000 مليار",
    "100000000000-count-few";
    "000 مليار",
    "100000000000-count-many";
    "000 مليار",
    "100000000000-count-other";
    "000 مليار",
    "1000000000000-count-zero";

```

```

        "0 تريليون",
        "1000000000000-count-one";
        "0 تريليون",
        "1000000000000-count-two";
        "0 تريليون",
        "1000000000000-count-few";
        "0 تريليونات",
        "1000000000000-count-many";
        "0 تريليون",
        "1000000000000-count-other";
        "0 تريليون",
        "1000000000000-count-zero";
        "00 تريليون",
        "10000000000000-count-one";
        "00 تريليون",
        "10000000000000-count-two";
        "00 تريليون",
        "10000000000000-count-few";
        "00 تريليون",
        "10000000000000-count-many";
        "00 تريليون",
        "10000000000000-count-other";
        "00 تريليون",
        "10000000000000-count-zero";
        "000 تريليون",
        "100000000000000-count-one";
        "000 تريليون",
        "100000000000000-count-two";
        "000 تريليون",
        "100000000000000-count-few";
        "000 تريليون",
        "100000000000000-count-many";
        "000 تريليون",
        "100000000000000-count-other";
        "000 تريليون";
    }
}
"short";
{
    "decimalFormat";
    {
        "1000-count-zero";
        "0 ألف",
        "1000-count-one";
        "0 ألف",
        "1000-count-two";
        "0 ألف",
        "1000-count-few";
        "0 آلاف",
        "1000-count-many";
        "0 ألف",
        "1000-count-other";
        "0 ألف",
        "10000-count-zero";
        "00 ألف",
        "10000-count-one";
        "00 ألف",

```

```
"10000-count-two";
"00 ألف",
"10000-count-few";
"00 ألف",
"10000-count-many";
"00 ألف",
"10000-count-other";
"00 ألف",
"100000-count-zero";
"000 ألف",
"100000-count-one";
"000 ألف",
"100000-count-two";
"000 ألف",
"100000-count-few";
"000 ألف",
"100000-count-many";
"000 ألف",
"100000-count-other";
"000 ألف",
"1000000-count-zero";
"0 مليون",
"1000000-count-one";
"0 مليون",
"1000000-count-two";
"0 مليون",
"1000000-count-few";
"0 مليون",
"1000000-count-many";
"0 مليون",
"1000000-count-other";
"0 مليون",
"10000000-count-zero";
"00 مليون",
"10000000-count-one";
"00 مليون",
"10000000-count-two";
"00 مليون",
"10000000-count-few";
"00 مليون",
"10000000-count-many";
"00 مليون",
"10000000-count-other";
"00 مليون",
"100000000-count-zero";
"000 مليون",
"100000000-count-one";
"000 مليون",
"100000000-count-two";
"000 مليون",
"100000000-count-few";
"000 مليون",
"100000000-count-many";
"000 مليون",
"100000000-count-other";
"000 مليون",
"1000000000-count-zero";
```

```
"0 مليا",
  "1000000000-count-one";
"0 مليا",
  "1000000000-count-two";
"0 مليا",
  "1000000000-count-few";
"0 مليا",
  "1000000000-count-many";
"0 مليا",
  "1000000000-count-other";
"0 مليا",
  "1000000000-count-zero";
"00 مليا",
  "10000000000-count-one";
"00 مليا",
  "10000000000-count-two";
"00 مليا",
  "10000000000-count-few";
"00 مليا",
  "10000000000-count-many";
"00 مليا",
  "10000000000-count-other";
"00 مليا",
  "10000000000-count-zero";
"000 مليا",
  "100000000000-count-one";
"000 مليا",
  "100000000000-count-two";
"000 مليا",
  "100000000000-count-few";
"000 مليا",
  "100000000000-count-many";
"000 مليا",
  "100000000000-count-other";
"000 مليا",
  "100000000000-count-zero";
"0 ترليو",
  "1000000000000-count-one";
"0 ترليو",
  "1000000000000-count-two";
"0 ترليو",
  "1000000000000-count-few";
"0 ترليو",
  "1000000000000-count-many";
"0 ترليو",
  "1000000000000-count-other";
"0 ترليو",
  "1000000000000-count-zero";
"00 ترليو",
  "10000000000000-count-one";
"00 ترليو",
  "10000000000000-count-two";
"00 ترليو",
  "10000000000000-count-few";
"00 ترليو",
  "10000000000000-count-many";
"00 ترليو",
```

```

        "10000000000000-count-other";
        "00 ترلیو",
        "10000000000000-count-zero";
        "000 ترلیو",
        "10000000000000-count-one";
        "000 ترلیو",
        "10000000000000-count-two";
        "000 ترلیو",
        "10000000000000-count-few";
        "000 ترلیو",
        "10000000000000-count-many";
        "000 ترلیو",
        "10000000000000-count-other";
        "000 ترلیو";
    }
}
}
"scientificFormats-numberSystem-arab";
{
    "standard";
    "#E0";
}
"scientificFormats-numberSystem-latn";
{
    "standard";
    "#E0";
}
"percentFormats-numberSystem-arab";
{
    "standard";
    "#,##0 %";
}
"percentFormats-numberSystem-latn";
{
    "standard";
    "#,##0%";
}
"currencyFormats-numberSystem-arab";
{
    "currencySpacing";
    {
        "beforeCurrency";
        {
            "currencyMatch";
            "[:^S:]",
            "surroundingMatch";
            "[:digit:]",
            "insertBetween";
            " ";
        }
        "afterCurrency";
        {
            "currencyMatch";
            "[:^S:]",
            "surroundingMatch";
            "[:digit:]",
            "insertBetween";

```

```

        " ";
    }
}
"standard";
"#,##0.00 ¤",
    "accounting";
"#,##0.00 ¤",
    "unitPattern-count-zero";
"{0} {1}",
    "unitPattern-count-one";
"{0} {1}",
    "unitPattern-count-two";
"{0} {1}",
    "unitPattern-count-few";
"{0} {1}",
    "unitPattern-count-many";
"{0} {1}",
    "unitPattern-count-other";
"{0} {1}";
}
"currencyFormats-numberSystem-latn";
{
    "currencySpacing";
    {
        "beforeCurrency";
        {
            "currencyMatch";
            "[:^S:]",
            "surroundingMatch";
            "[:digit:]",
            "insertBetween";
            " ";
        }
        "afterCurrency";
        {
            "currencyMatch";
            "[:^S:]",
            "surroundingMatch";
            "[:digit:]",
            "insertBetween";
            " ";
        }
    }
}
"standard";
"¤ #,##0.00",
    "accounting";
"¤#,##0.00; (¤#,##0.00) ",
    "short";
{
    "standard";
    {
        "1000-count-zero";
        "¤ 0 ألف",
        "1000-count-one";
        "¤ 0 ألف",
        "1000-count-two";
        "¤ 0 ألف",
    }
}

```

```

        "1000-count-few";
    "ألف 0",
        "1000-count-many";
    "ألف 0",
        "1000-count-other";
    "ألف 0",
        "10000-count-zero";
    "ألف 00",
        "10000-count-one";
    "ألف 00",
        "10000-count-two";
    "ألف 00",
        "10000-count-few";
    "ألف 00",
        "10000-count-many";
    "ألف 00",
        "10000-count-other";
    "ألف 00",
        "100000-count-zero";
    "ألف 000",
        "100000-count-one";
    "ألف 000",
        "100000-count-two";
    "ألف 000",
        "100000-count-few";
    "ألف 000",
        "100000-count-many";
    "ألف 000",
        "100000-count-other";
    "ألف 000",
        "1000000-count-zero";
    "مليون 0",
        "1000000-count-one";
    "مليون 0",
        "1000000-count-two";
    "مليون 0",
        "1000000-count-few";
    "مليون 0",
        "1000000-count-many";
    "مليون 0",
        "1000000-count-other";
    "مليون 0",
        "10000000-count-zero";
    "مليون 00",
        "10000000-count-one";
    "مليون 00",
        "10000000-count-two";
    "مليون 00",
        "10000000-count-few";
    "مليون 00",
        "10000000-count-many";
    "مليون 00",
        "10000000-count-other";
    "مليون 00",
        "100000000-count-zero";
    "مليون 000",
        "100000000-count-one";

```

```
"۰ 000 مليو",
    "100000000-count-two";
"۰ 000 مليو",
    "100000000-count-few";
"۰ 000 مليو",
    "100000000-count-many";
"۰ 000 مليو",
    "100000000-count-other";
"۰ 000 مليو",
    "100000000-count-zero";
"۰ 0 مليا",
    "100000000-count-one";
"۰ 0 مليا",
    "100000000-count-two";
"۰ 0 مليا",
    "100000000-count-few";
"۰ 0 مليا",
    "100000000-count-many";
"۰ 0 مليا",
    "100000000-count-other";
"۰ 0 مليا",
    "100000000-count-zero";
"۰ 00 مليا",
    "100000000-count-one";
"۰ 00 مليا",
    "100000000-count-two";
"۰ 00 مليا",
    "100000000-count-few";
"۰ 00 مليا",
    "100000000-count-many";
"۰ 00 مليا",
    "100000000-count-other";
"۰ 00 مليا",
    "100000000-count-zero";
"۰ 000 مليا",
    "100000000-count-one";
"۰ 000 مليا",
    "100000000-count-two";
"۰ 000 مليا",
    "100000000-count-few";
"۰ 000 مليا",
    "100000000-count-many";
"۰ 000 مليا",
    "100000000-count-other";
"۰ 000 مليا",
    "100000000-count-zero";
"۰ 0 ترليو",
    "100000000-count-one";
"۰ 0 ترليو",
    "100000000-count-two";
"۰ 0 ترليو",
    "100000000-count-few";
"۰ 0 ترليو",
    "100000000-count-many";
"۰ 0 ترليو",
    "100000000-count-other";
"۰ 0 ترليو",
```



```

        "10000000000000-count-zero";
        "ترليو",
        "10000000000000-count-one";
        "ترليو",
        "10000000000000-count-two";
        "ترليو",
        "10000000000000-count-few";
        "ترليو",
        "10000000000000-count-many";
        "ترليو",
        "10000000000000-count-other";
        "ترليو",
        "10000000000000-count-zero";
        "ترليو",
        "10000000000000-count-one";
        "ترليو",
        "10000000000000-count-two";
        "ترليو",
        "10000000000000-count-few";
        "ترليو",
        "10000000000000-count-many";
        "ترليو",
        "10000000000000-count-other";
        "ترليو";
    }
}

unitPattern-count-zero";
"{0} {1}",
    unitPattern-count-one";
"{0} {1}",
    unitPattern-count-two";
"{0} {1}",
    unitPattern-count-few";
"{0} {1}",
    unitPattern-count-many";
"{0} {1}",
    unitPattern-count-other";
"{0} {1}";
}

miscPatterns-numberSystem-arab";
{
    atLeast";
    "+{0}",
    range";
    "{0}-{1}";
}

miscPatterns-numberSystem-latn";
{
    atLeast";
    "+{0}",
    range";
    "{0}-{1}";
}
}
}

```

TIMEZONENAMES.JSON

```
{
  "main": {
    "ar": {
      "identity": {
        "version": {
          "_number": "$Revision: 12879 $",
          "_cldrVersion": "30.0.3"
        },
        "language": "ar"
      },
      "dates": {
        "timeZoneNames": {
          "hourFormat": "+HH:mm;-HH:mm",
          "gmtFormat": "0{جرينتش",
          "gmtZeroFormat": "جرينتش",
          "regionFormat": "0{توقيت",
          "regionFormat-type-daylight": "0{الصيفي",
          "regionFormat-type-standard": "0{الرسمي",
          "fallbackFormat": "{1} ({0})",
          "zone": {
            "America": {
              "Adak": {
                "exemplarCity": "أداك"
              },
              "Anchorage": {
                "exemplarCity": "أنشوراج"
              },
              "Anguilla": {
                "exemplarCity": "أنغيلا"
              },
              "Antigua": {
                "exemplarCity": "أنتيغوا"
              },
              "Araguaina": {
                "exemplarCity": "أروجوانيا"
              },
              "Argentina": {
                "Rio_Gallegos": {
                  "exemplarCity": "ريو جالييوس"
                },
                "San_Juan": {
                  "exemplarCity": "سان خوان"
                },
                "Ushuaia": {
                  "exemplarCity": "أشوا"
                },
                "La_Rioja": {
                  "exemplarCity": "لا ريوجا"
                },
                "San_Luis": {
                  "exemplarCity": "سان لويس"
                },
                "Salta": {
```

```

        "exemplarCity": "سالتا"
    },
    "Tucuman": {
        "exemplarCity": "تاکمان"
    }
},
"Aruba": {
    "exemplarCity": "أروبا"
},
"Asuncion": {
    "exemplarCity": "أسونسيون"
},
"Bahia": {
    "exemplarCity": "باهيا"
},
"Bahia_Banderas": {
    "exemplarCity": "باهيا بانديراس"
},
"Barbados": {
    "exemplarCity": "بربادوس"
},
"Belem": {
    "exemplarCity": "بلم"
},
"Belize": {
    "exemplarCity": "بليز"
},
"Blanc-Sablon": {
    "exemplarCity": "بلانك-سابلون"
},
"Boa_Vista": {
    "exemplarCity": "باو فيستا"
},
"Bogota": {
    "exemplarCity": "بوغوتا"
},
"Boise": {
    "exemplarCity": "بويس"
},
"Buenos_Aires": {
    "exemplarCity": "بوينوس آيرس"
},
"Cambridge_Bay": {
    "exemplarCity": "كامبرديج باي"
},
"Campo_Grande": {
    "exemplarCity": "كومبو جراند"
},
"Cancun": {
    "exemplarCity": "كانكون"
},
"Caracas": {
    "exemplarCity": "كاراكاس"
},
"Catamarca": {
    "exemplarCity": "كاتاماركا"
},

```

```
"Cayenne": {
  "exemplarCity": "كايين"
},
"Cayman": {
  "exemplarCity": "كيمان"
},
"Chicago": {
  "exemplarCity": "شيكاغو"
},
"Chihuahua": {
  "exemplarCity": "تشيوأوا"
},
"Coral_Harbour": {
  "exemplarCity": "كورال هاربر"
},
"Cordoba": {
  "exemplarCity": "كوردوبا"
},
"Costa_Rica": {
  "exemplarCity": "كوستاريكا"
},
"Creston": {
  "exemplarCity": "كريستون"
},
"Cuiaba": {
  "exemplarCity": "كيابا"
},
"Curacao": {
  "exemplarCity": "كوراكاو"
},
"Danmarkshavn": {
  "exemplarCity": "دانمرك شافن"
},
"Dawson": {
  "exemplarCity": "داوسان"
},
"Dawson_Creek": {
  "exemplarCity": "داوسن كريك"
},
"Denver": {
  "exemplarCity": "دنفر"
},
"Detroit": {
  "exemplarCity": "ديترويت"
},
"Dominica": {
  "exemplarCity": "دومينيكا"
},
"Edmonton": {
  "exemplarCity": "ايدمونتون"
},
"Eirunepe": {
  "exemplarCity": "ايرونبي"
},
"El_Salvador": {
  "exemplarCity": "السلفادور"
},
},
```

```

"Fort_Nelson": {
  "exemplarCity": "فورت نيلسون"
},
"Fortaleza": {
  "exemplarCity": "فورتاليزا"
},
"Glace_Bay": {
  "exemplarCity": "جلاس باي"
},
"Godthab": {
  "exemplarCity": "غودثاب"
},
"Goose_Bay": {
  "exemplarCity": "جوس باي"
},
"Grand_Turk": {
  "exemplarCity": "غراند ترك"
},
"Grenada": {
  "exemplarCity": "جرينادا"
},
"Guadeloupe": {
  "exemplarCity": "غوادلوب"
},
"Guatemala": {
  "exemplarCity": "غواتيمالا"
},
"Guayaquil": {
  "exemplarCity": "غواياكويل"
},
"Guyana": {
  "exemplarCity": "غيانا"
},
"Halifax": {
  "exemplarCity": "هاليفاكس"
},
"Havana": {
  "exemplarCity": "هافانا"
},
"Hermosillo": {
  "exemplarCity": "هيرموسيلو"
},
"Indiana": {
  "Vincennes": {
    "exemplarCity": "فينسينس"
  },
  "Petersburg": {
    "exemplarCity": "بيتربيرغ"
  },
  "Tell_City": {
    "exemplarCity": "مدينة تل، إنديانا"
  },
  "Knox": {
    "exemplarCity": "كونكس"
  },
  "Winamac": {
    "exemplarCity": "ويناماك"
  }
}

```

```

    },
    "Marengo": {
      "exemplarCity": "مارنجو"
    },
    "Vevay": {
      "exemplarCity": "فيفاي"
    }
  },
  "Indianapolis": {
    "exemplarCity": "إنديانا بوليس"
  },
  "Inuvik": {
    "exemplarCity": "اينوفيك"
  },
  "Iqaluit": {
    "exemplarCity": "اكويلت"
  },
  "Jamaica": {
    "exemplarCity": "جامايكا"
  },
  "Jujuy": {
    "exemplarCity": "جوجو"
  },
  "Juneau": {
    "exemplarCity": "جونى"
  },
  "Kentucky": {
    "Monticello": {
      "exemplarCity": "مونتي سيلو"
    }
  },
  "Kralendijk": {
    "exemplarCity": "كرالنديك"
  },
  "La_Paz": {
    "exemplarCity": "لا باز"
  },
  "Lima": {
    "exemplarCity": "ليما"
  },
  "Los_Angeles": {
    "exemplarCity": "لوس انجلوس"
  },
  "Louisville": {
    "exemplarCity": "لويس فيل"
  },
  "Lower_Princes": {
    "exemplarCity": "حي الأمير السفلي"
  },
  "Maceio": {
    "exemplarCity": "ماشيو"
  },
  "Managua": {
    "exemplarCity": "ماناغوا"
  },
  "Manaus": {
    "exemplarCity": "ماناوس"
  }

```

```
,
"Marigot": {
  "exemplarCity": "ماريغوت"
},
"Martinique": {
  "exemplarCity": "المارتينيك"
},
"Matamoros": {
  "exemplarCity": "ماتاموروس"
},
"Mazatlan": {
  "exemplarCity": "مازااتلان"
},
"Mendoza": {
  "exemplarCity": "ميندوزا"
},
"Menominee": {
  "exemplarCity": "مينوميني"
},
"Merida": {
  "exemplarCity": "ميريديا"
},
"Metlakatla": {
  "exemplarCity": "ميتلاكاتلا"
},
"Mexico_City": {
  "exemplarCity": "مدينة المكسيك"
},
"Miquelon": {
  "exemplarCity": "ميكلون"
},
"Moncton": {
  "exemplarCity": "وينكتون"
},
"Monterrey": {
  "exemplarCity": "مونتيري"
},
"Montevideo": {
  "exemplarCity": "مونتيفيديو"
},
"Montserrat": {
  "exemplarCity": "مونتسيرات"
},
"Nassau": {
  "exemplarCity": "ناسو"
},
"New_York": {
  "exemplarCity": "نيويورك"
},
"Nipigon": {
  "exemplarCity": "نيبيجون"
},
"Nome": {
  "exemplarCity": "نوم"
},
"Noronha": {
  "exemplarCity": "نوروناه"
```

```

    },
    "North_Dakota": {
      "Beulah": {
        "exemplarCity": "بيولا، داكوتا الشمالية"
      },
      "New_Salem": {
        "exemplarCity": "نيو ساليم"
      },
      "Center": {
        "exemplarCity": "سنتر"
      }
    },
    "Ojinaga": {
      "exemplarCity": "أوجيناغا"
    },
    "Panama": {
      "exemplarCity": "بنما"
    },
    "Pangnirtung": {
      "exemplarCity": "بانجينتينج"
    },
    "Paramaribo": {
      "exemplarCity": "باراماريبو"
    },
    "Phoenix": {
      "exemplarCity": "فينكس"
    },
    "Port-au-Prince": {
      "exemplarCity": "بورت أو برنس"
    },
    "Port_of_Spain": {
      "exemplarCity": "بورت أوف سبين"
    },
    "Porto_Velho": {
      "exemplarCity": "بورتو فيلو"
    },
    "Puerto_Rico": {
      "exemplarCity": "بورتوريكو"
    },
    "Rainy_River": {
      "exemplarCity": "راني ريفر"
    },
    "Rankin_Inlet": {
      "exemplarCity": "رانكن انلت"
    },
    "Recife": {
      "exemplarCity": "ريسيف"
    },
    "Regina": {
      "exemplarCity": "ريجينا"
    },
    "Resolute": {
      "exemplarCity": "ريزولوت"
    },
    "Rio_Branco": {
      "exemplarCity": "ريوبرانكو"
    },
  },

```



```

"Santa_Isabel": {
  "exemplarCity": "سانتا إيزابيل"
},
"Santarem": {
  "exemplarCity": "سانتاريم"
},
"Santiago": {
  "exemplarCity": "سانتياغو"
},
"Santo_Domingo": {
  "exemplarCity": "سانتو دومينغو"
},
"Sao_Paulo": {
  "exemplarCity": "ساو باولو"
},
"Scoresbysund": {
  "exemplarCity": "سكورسبيسند"
},
"Sitka": {
  "exemplarCity": "سيتكا"
},
"St_Barthelemy": {
  "exemplarCity": "سانت بارتيليمي"
},
"St_Johns": {
  "exemplarCity": "سانت جونز"
},
"St_Kitts": {
  "exemplarCity": "سانت كيتس"
},
"St_Lucia": {
  "exemplarCity": "سانت لوشيا"
},
"St_Thomas": {
  "exemplarCity": "سانت توماس"
},
"St_Vincent": {
  "exemplarCity": "سانت فنسنت"
},
"Swift_Current": {
  "exemplarCity": "سوفت كارنت"
},
"Tegucigalpa": {
  "exemplarCity": "تيغوسيغالبا"
},
"Thule": {
  "exemplarCity": "ثيل"
},
"Thunder_Bay": {
  "exemplarCity": "ثندر باي"
},
"Tijuana": {
  "exemplarCity": "تيخوانا"
},
"Toronto": {
  "exemplarCity": "تورونتو"
},

```

```

    "Tortola": {
      "exemplarCity": "تورتولا"
    },
    "Vancouver": {
      "exemplarCity": "فانكوفر"
    },
    "Whitehorse": {
      "exemplarCity": "وايت هورس"
    },
    "Winnipeg": {
      "exemplarCity": "وينيبج"
    },
    "Yakutat": {
      "exemplarCity": "ياكوتات"
    },
    "Yellowknife": {
      "exemplarCity": "يلونيف"
    }
  },
  "Atlantic": {
    "Azores": {
      "exemplarCity": "أزورس"
    },
    "Bermuda": {
      "exemplarCity": "برمودا"
    },
    "Canary": {
      "exemplarCity": "كناري"
    },
    "Cape_Verde": {
      "exemplarCity": "الرأس الأخضر"
    },
    "Faeroe": {
      "exemplarCity": "فارو"
    },
    "Madeira": {
      "exemplarCity": "ماديرا"
    },
    "Reykjavik": {
      "exemplarCity": "ريكيافيك"
    },
    "South_Georgia": {
      "exemplarCity": "جورجيا الجنوبية"
    },
    "St_Helena": {
      "exemplarCity": "سانت هيلينا"
    },
    "Stanley": {
      "exemplarCity": "استانلي"
    }
  },
  "Europe": {
    "Amsterdam": {
      "exemplarCity": "أمستردام"
    },
    "Andorra": {
      "exemplarCity": "أندورا"
    }
  }
}

```

```

    },
    "Astrakhan": {
      "exemplarCity": "أستراخان"
    },
    "Athens": {
      "exemplarCity": "أثينا"
    },
    "Belgrade": {
      "exemplarCity": "بلغراد"
    },
    "Berlin": {
      "exemplarCity": "برلين"
    },
    "Bratislava": {
      "exemplarCity": "براتيسلافا"
    },
    "Brussels": {
      "exemplarCity": "بروكسل"
    },
    "Bucharest": {
      "exemplarCity": "بوخارست"
    },
    "Budapest": {
      "exemplarCity": "بودابست"
    },
    "Busingen": {
      "exemplarCity": "بوسنغن"
    },
    "Chisinau": {
      "exemplarCity": "تشيسيناو"
    },
    "Copenhagen": {
      "exemplarCity": "كوبنهاغن"
    },
    "Dublin": {
      "long": {
        "daylight": "توقيت أيرلندا الرسمي"
      },
      "exemplarCity": "دبلن"
    },
    "Gibraltar": {
      "exemplarCity": "جبل طارق"
    },
    "Guernsey": {
      "exemplarCity": "غيرنسي"
    },
    "Helsinki": {
      "exemplarCity": "هلسنكي"
    },
    "Isle_of_Man": {
      "exemplarCity": "جزيرة مان"
    },
    "Istanbul": {
      "exemplarCity": "إسطنبول"
    },
    "Jersey": {
      "exemplarCity": "جيرسي"
    }
  }

```

```
},
"Kalininingrad": {
  "exemplarCity": "كالينجراد"
},
"Kiev": {
  "exemplarCity": "كييف"
},
"Kirov": {
  "exemplarCity": "كيروف"
},
"Lisbon": {
  "exemplarCity": "لشبونة"
},
"Ljubljana": {
  "exemplarCity": "ليوبليانا"
},
"London": {
  "long": {
    "daylight": "توقيت بريطانيا الصيفي"
  },
  "exemplarCity": "لندن"
},
"Luxembourg": {
  "exemplarCity": "لوكسمبورغ"
},
"Madrid": {
  "exemplarCity": "مدريد"
},
"Malta": {
  "exemplarCity": "مالطة"
},
"Mariehamn": {
  "exemplarCity": "ماريهامن"
},
"Minsk": {
  "exemplarCity": "مينسك"
},
"Monaco": {
  "exemplarCity": "موناكو"
},
"Moscow": {
  "exemplarCity": "موسكو"
},
"Oslo": {
  "exemplarCity": "أوسلو"
},
"Paris": {
  "exemplarCity": "باريس"
},
"Podgorica": {
  "exemplarCity": "بودغوريكا"
},
"Prague": {
  "exemplarCity": "براغ"
},
"Riga": {
  "exemplarCity": "ريغا"
}
```

```
    },  
    "Rome": {  
      "exemplarCity": "روما"  
    },  
    "Samara": {  
      "exemplarCity": "سمراء"  
    },  
    "San_Marino": {  
      "exemplarCity": "سان مارينو"  
    },  
    "Sarajevo": {  
      "exemplarCity": "سراييفو"  
    },  
    "Simferopol": {  
      "exemplarCity": "سيمفروبول"  
    },  
    "Skopje": {  
      "exemplarCity": "سكوبي"  
    },  
    "Sofia": {  
      "exemplarCity": "صوفيا"  
    },  
    "Stockholm": {  
      "exemplarCity": "ستوكهولم"  
    },  
    "Tallinn": {  
      "exemplarCity": "تالين"  
    },  
    "Tirane": {  
      "exemplarCity": "تيرانا"  
    },  
    "Ulyanovsk": {  
      "exemplarCity": "أوليانوفسك"  
    },  
    "Uzhgorod": {  
      "exemplarCity": "أوزجروود"  
    },  
    "Vaduz": {  
      "exemplarCity": "فادوز"  
    },  
    "Vatican": {  
      "exemplarCity": "الفاتيكان"  
    },  
    "Vienna": {  
      "exemplarCity": "فيينا"  
    },  
    "Vilnius": {  
      "exemplarCity": "فيلنيوس"  
    },  
    "Volgograd": {  
      "exemplarCity": "فولجوراد"  
    },  
    "Warsaw": {  
      "exemplarCity": "وارسو"  
    },  
    "Zagreb": {  
      "exemplarCity": "زغرب"
```

```
    },
    "Zaporozhye": {
      "exemplarCity": "زابوروزي"
    },
    "Zurich": {
      "exemplarCity": "زيورخ"
    }
  },
  "Africa": {
    "Abidjan": {
      "exemplarCity": "أبيدجان"
    },
    "Accra": {
      "exemplarCity": "أكرا"
    },
    "Addis_Ababa": {
      "exemplarCity": "أديس أبابا"
    },
    "Algiers": {
      "exemplarCity": "الجزائر"
    },
    "Asmera": {
      "exemplarCity": "أسمره"
    },
    "Bamako": {
      "exemplarCity": "باماكو"
    },
    "Bangui": {
      "exemplarCity": "بانغوي"
    },
    "Banjul": {
      "exemplarCity": "بانجول"
    },
    "Bissau": {
      "exemplarCity": "بيساو"
    },
    "Blantyre": {
      "exemplarCity": "بلانتيير"
    },
    "Brazzaville": {
      "exemplarCity": "برازافيل"
    },
    "Bujumbura": {
      "exemplarCity": "بوجومبورا"
    },
    "Cairo": {
      "exemplarCity": "القاهرة"
    },
    "Casablanca": {
      "exemplarCity": "الدار البيضاء"
    },
    "Ceuta": {
      "exemplarCity": "سيتا"
    },
    "Conakry": {
      "exemplarCity": "كوناكري"
    }
  },
}
```

```
"Dakar": {
  "exemplarCity": "داكار"
},
"Dar_es_Salaam": {
  "exemplarCity": "دار السلام"
},
"Djibouti": {
  "exemplarCity": "جيبوتي"
},
"Douala": {
  "exemplarCity": "دوالا"
},
"El_Aaiun": {
  "exemplarCity": "العيون"
},
"Freetown": {
  "exemplarCity": "فري تاون"
},
"Gaborone": {
  "exemplarCity": "غابورون"
},
"Harare": {
  "exemplarCity": "هراري"
},
"Johannesburg": {
  "exemplarCity": "جوهانسبرغ"
},
"Juba": {
  "exemplarCity": "جوبا"
},
"Kampala": {
  "exemplarCity": "كامبالا"
},
"Khartoum": {
  "exemplarCity": "الخرطوم"
},
"Kigali": {
  "exemplarCity": "كيغالي"
},
"Kinshasa": {
  "exemplarCity": "كينشاسا"
},
"Lagos": {
  "exemplarCity": "لاغوس"
},
"Libreville": {
  "exemplarCity": "ليبرفيل"
},
"Lome": {
  "exemplarCity": "لومي"
},
"Luanda": {
  "exemplarCity": "لواندا"
},
"Lubumbashi": {
  "exemplarCity": "لومبباشا"
},
}
```

```

    "Lusaka": {
      "exemplarCity": "لوساكا"
    },
    "Malabo": {
      "exemplarCity": "مالابو"
    },
    "Maputo": {
      "exemplarCity": "مابوتو"
    },
    "Maseru": {
      "exemplarCity": "ماسيرو"
    },
    "Mbabane": {
      "exemplarCity": "مباباني"
    },
    "Mogadishu": {
      "exemplarCity": "مقديشو"
    },
    "Monrovia": {
      "exemplarCity": "مونروفيا"
    },
    "Nairobi": {
      "exemplarCity": "نيروبي"
    },
    "Ndjamena": {
      "exemplarCity": "نجامينا"
    },
    "Niamey": {
      "exemplarCity": "نيامي"
    },
    "Nouakchott": {
      "exemplarCity": "نواكشوط"
    },
    "Ouagadougou": {
      "exemplarCity": "واغادوغو"
    },
    "Porto-Novo": {
      "exemplarCity": "بورتو نوفو"
    },
    "Sao_Tome": {
      "exemplarCity": "ساو تومي"
    },
    "Tripoli": {
      "exemplarCity": "طرابلس"
    },
    "Tunis": {
      "exemplarCity": "تونس"
    },
    "Windhoek": {
      "exemplarCity": "ويندهوك"
    }
  },
  "Asia": {
    "Aden": {
      "exemplarCity": "عدن"
    },
    "Almaty": {

```



```
    "exemplarCity": "ألماتي"
  },
  "Amman": {
    "exemplarCity": "عمان"
  },
  "Anadyr": {
    "exemplarCity": "أندير"
  },
  "Aqtau": {
    "exemplarCity": "أكتاو"
  },
  "Aqtobe": {
    "exemplarCity": "أکتوب"
  },
  "Ashgabat": {
    "exemplarCity": "عشق آباد"
  },
  "Baghdad": {
    "exemplarCity": "بغداد"
  },
  "Bahrain": {
    "exemplarCity": "البحرين"
  },
  "Baku": {
    "exemplarCity": "باکو"
  },
  "Bangkok": {
    "exemplarCity": "بانكوك"
  },
  "Barnaul": {
    "exemplarCity": "بارناول"
  },
  "Beirut": {
    "exemplarCity": "بيروت"
  },
  "Bishkek": {
    "exemplarCity": "بشكيك"
  },
  "Brunei": {
    "exemplarCity": "بروناي"
  },
  "Calcutta": {
    "exemplarCity": "كالكتا"
  },
  "Chita": {
    "exemplarCity": "تشيتا"
  },
  "Choibalsan": {
    "exemplarCity": "تشوبالسان"
  },
  "Colombo": {
    "exemplarCity": "كولومبو"
  },
  "Damascus": {
    "exemplarCity": "دمشق"
  },
  "Dhaka": {
```

```

    "exemplarCity": "دكا"
  },
  "Dili": {
    "exemplarCity": "دیلی"
  },
  "Dubai": {
    "exemplarCity": "دبی"
  },
  "Dushanbe": {
    "exemplarCity": "دوشانبی"
  },
  "Gaza": {
    "exemplarCity": "غزة"
  },
  "Hebron": {
    "exemplarCity": "هیبرون (مدینة الخلیل)"
  },
  "Hong_Kong": {
    "exemplarCity": "هونغ كونغ"
  },
  "Hovd": {
    "exemplarCity": "هوفد"
  },
  "Irkutsk": {
    "exemplarCity": "ایرکییتسک"
  },
  "Jakarta": {
    "exemplarCity": "جاکرتا"
  },
  "Jayapura": {
    "exemplarCity": "جایابورا"
  },
  "Jerusalem": {
    "exemplarCity": "القدس"
  },
  "Kabul": {
    "exemplarCity": "کابل"
  },
  "Kamchatka": {
    "exemplarCity": "کامتشاتکا"
  },
  "Karachi": {
    "exemplarCity": "کراتھی"
  },
  "Katmandu": {
    "exemplarCity": "کاتماندو"
  },
  "Khandyga": {
    "exemplarCity": "خاندیجا"
  },
  "Krasnoyarsk": {
    "exemplarCity": "کراسنویارسک"
  },
  "Kuala_Lumpur": {
    "exemplarCity": "کوالا لامپور"
  },
  "Kuching": {

```

```

    "exemplarCity": "كيشينج"
  },
  "Kuwait": {
    "exemplarCity": "الكويت"
  },
  "Macau": {
    "exemplarCity": "ماكاو"
  },
  "Magadan": {
    "exemplarCity": "مجادن"
  },
  "Makassar": {
    "exemplarCity": "ماكسار"
  },
  "Manila": {
    "exemplarCity": "مانيلا"
  },
  "Muscat": {
    "exemplarCity": "مسقط"
  },
  "Nicosia": {
    "exemplarCity": "نيقوسيا"
  },
  "Novokuznetsk": {
    "exemplarCity": "نوفوكوزنتسك"
  },
  "Novosibirsk": {
    "exemplarCity": "نوفوسبيرسك"
  },
  "Omsk": {
    "exemplarCity": "أومسك"
  },
  "Oral": {
    "exemplarCity": "أورال"
  },
  "Phnom_Penh": {
    "exemplarCity": "بنوم بنه"
  },
  "Pontianak": {
    "exemplarCity": "بونتيانك"
  },
  "Pyongyang": {
    "exemplarCity": "بيونغ يانغ"
  },
  "Qatar": {
    "exemplarCity": "قطر"
  },
  "Qyzylorda": {
    "exemplarCity": "كيزيلوردا"
  },
  "Rangoon": {
    "exemplarCity": "رانغون"
  },
  "Riyadh": {
    "exemplarCity": "الرياض"
  },
  "Saigon": {

```

```
    "exemplarCity": "مدينة هو تشي منه",
  },
  "Sakhalin": {
    "exemplarCity": "سكالين",
  },
  "Samarkand": {
    "exemplarCity": "سمرقند",
  },
  "Seoul": {
    "exemplarCity": "سول",
  },
  "Shanghai": {
    "exemplarCity": "شنغهاي",
  },
  "Singapore": {
    "exemplarCity": "سنغافورة",
  },
  "Srednekolymsk": {
    "exemplarCity": "سريدنكوليمسك",
  },
  "Taipei": {
    "exemplarCity": "تايبه",
  },
  "Tashkent": {
    "exemplarCity": "طشقند",
  },
  "Tbilisi": {
    "exemplarCity": "تبليسي",
  },
  "Tehran": {
    "exemplarCity": "طهران",
  },
  "Thimphu": {
    "exemplarCity": "تيمفو",
  },
  "Tokyo": {
    "exemplarCity": "طوكيو",
  },
  "Tomsk": {
    "exemplarCity": "تومسك",
  },
  "Ulaanbaatar": {
    "exemplarCity": "أولانباتار",
  },
  "Urumqi": {
    "exemplarCity": "أرومكي",
  },
  "Ust-Nera": {
    "exemplarCity": "أوست نيرا",
  },
  "Vientiane": {
    "exemplarCity": "فيانتيان",
  },
  "Vladivostok": {
    "exemplarCity": "فلاديفوستك",
  },
  "Yakutsk": {
```

```

        "exemplarCity": "ياكتسك"
      },
      "Yekaterinburg": {
        "exemplarCity": "يكاترنبيرج"
      },
      "Yerevan": {
        "exemplarCity": "يريفان"
      }
    },
    "Indian": {
      "Antananarivo": {
        "exemplarCity": "أنتاناناريفو"
      },
      "Chagos": {
        "exemplarCity": "تشاغوس"
      },
      "Christmas": {
        "exemplarCity": "كريسماس"
      },
      "Cocos": {
        "exemplarCity": "كوكوس"
      },
      "Comoro": {
        "exemplarCity": "جزر القمر"
      },
      "Kerguelen": {
        "exemplarCity": "كيرغويلين"
      },
      "Mahe": {
        "exemplarCity": "ماهي"
      },
      "Maldives": {
        "exemplarCity": "المالديف"
      },
      "Mauritius": {
        "exemplarCity": "موريشيوس"
      },
      "Mayotte": {
        "exemplarCity": "مايوت"
      },
      "Reunion": {
        "exemplarCity": "ريونيون"
      }
    },
    "Australia": {
      "Adelaide": {
        "exemplarCity": "أديليد"
      },
      "Brisbane": {
        "exemplarCity": "برسبان"
      },
      "Broken_Hill": {
        "exemplarCity": "بروكن هيل"
      },
      "Currie": {
        "exemplarCity": "كوري"
      }
    }
  },

```

```
"Darwin": {
  "exemplarCity": "دارون"
},
"Eucla": {
  "exemplarCity": "أوكلا"
},
"Hobart": {
  "exemplarCity": "هوبارت"
},
"Lindeman": {
  "exemplarCity": "ليندمان"
},
"Lord_Howe": {
  "exemplarCity": "لورد هاو"
},
"Melbourne": {
  "exemplarCity": "ميلبورن"
},
"Perth": {
  "exemplarCity": "برثا"
},
"Sydney": {
  "exemplarCity": "سيدني"
}
},
"Pacific": {
  "Apia": {
    "exemplarCity": "أبيا"
  },
  "Auckland": {
    "exemplarCity": "أوكلاند"
  },
  "Bougainville": {
    "exemplarCity": "بوغانفيل"
  },
  "Chatham": {
    "exemplarCity": "تشاثام"
  },
  "Easter": {
    "exemplarCity": "استر"
  },
  "Efate": {
    "exemplarCity": "إيفات"
  },
  "Enderbury": {
    "exemplarCity": "اندربيرج"
  },
  "Fakaofu": {
    "exemplarCity": "فاكاوفو"
  },
  "Fiji": {
    "exemplarCity": "فيجي"
  },
  "Funafuti": {
    "exemplarCity": "فونافوتي"
  },
  "Galapagos": {
```

```

    "exemplarCity": "جلا باجوس"
  },
  "Gambier": {
    "exemplarCity": "جامبير"
  },
  "Guadalcanal": {
    "exemplarCity": "غواد الكانال"
  },
  "Guam": {
    "exemplarCity": "غوام"
  },
  "Honolulu": {
    "exemplarCity": "هونولولو"
  },
  "Johnston": {
    "exemplarCity": "جونستون"
  },
  "Kiritimati": {
    "exemplarCity": "كيريتي ماتي"
  },
  "Kosrae": {
    "exemplarCity": "كوسرا"
  },
  "Kwajalein": {
    "exemplarCity": "كواجالين"
  },
  "Majuro": {
    "exemplarCity": "ماجورو"
  },
  "Marquesas": {
    "exemplarCity": "ماركيساس"
  },
  "Midway": {
    "exemplarCity": "ميدواي"
  },
  "Nauru": {
    "exemplarCity": "ناورو"
  },
  "Niue": {
    "exemplarCity": "نيوي"
  },
  "Norfolk": {
    "exemplarCity": "نورفولك"
  },
  "Noumea": {
    "exemplarCity": "نوميا"
  },
  "Pago_Pago": {
    "exemplarCity": "باغو باغو"
  },
  "Palau": {
    "exemplarCity": "بالاو"
  },
  "Pitcairn": {
    "exemplarCity": "بيتكيرن"
  },
  "Ponape": {

```

```

        "exemplarCity": "باناب"
    },
    "Port_Moresby": {
        "exemplarCity": "بور مورسبي"
    },
    "Rarotonga": {
        "exemplarCity": "راروتونغا"
    },
    "Saipan": {
        "exemplarCity": "سايبان"
    },
    "Tahiti": {
        "exemplarCity": "تاهيتي"
    },
    "Tarawa": {
        "exemplarCity": "تاراوا"
    },
    "Tongatapu": {
        "exemplarCity": "تونغاتابو"
    },
    "Truk": {
        "exemplarCity": "ترك"
    },
    "Wake": {
        "exemplarCity": "واك"
    },
    "Wallis": {
        "exemplarCity": "واليس"
    }
},
"Arctic": {
    "Longyearbyen": {
        "exemplarCity": "لونغجيربين"
    }
},
"Antarctica": {
    "Casey": {
        "exemplarCity": "كاساي"
    },
    "Davis": {
        "exemplarCity": "دافيز"
    },
    "DumontDUrville": {
        "exemplarCity": "دي مونت دو روفيل"
    },
    "Macquarie": {
        "exemplarCity": "ماكوارى"
    },
    "Mawson": {
        "exemplarCity": "ماوسون"
    },
    "McMurdo": {
        "exemplarCity": "ماك موردو"
    },
    "Palmer": {
        "exemplarCity": "بالمير"
    }
},

```



```
"Rothera": {
  "exemplarCity": "روثيرا"
},
"Syowa": {
  "exemplarCity": "سايووا"
},
"Troll": {
  "exemplarCity": "ترول"
},
"Vostok": {
  "exemplarCity": "فوستوك"
}
},
"Etc": {
  "GMT": {
    "exemplarCity": "GMT"
  },
  "GMT1": {
    "exemplarCity": "GMT+1"
  },
  "GMT10": {
    "exemplarCity": "GMT+10"
  },
  "GMT11": {
    "exemplarCity": "GMT+11"
  },
  "GMT12": {
    "exemplarCity": "GMT+12"
  },
  "GMT2": {
    "exemplarCity": "GMT+2"
  },
  "GMT3": {
    "exemplarCity": "GMT+3"
  },
  "GMT4": {
    "exemplarCity": "GMT+4"
  },
  "GMT5": {
    "exemplarCity": "GMT+5"
  },
  "GMT6": {
    "exemplarCity": "GMT+6"
  },
  "GMT7": {
    "exemplarCity": "GMT+7"
  },
  "GMT8": {
    "exemplarCity": "GMT+8"
  },
  "GMT9": {
    "exemplarCity": "GMT+9"
  },
  "GMT-1": {
    "exemplarCity": "GMT-1"
  },
  "GMT-10": {
```

```

        "exemplarCity": "GMT-10"
      },
      "GMT-11": {
        "exemplarCity": "GMT-11"
      },
      "GMT-12": {
        "exemplarCity": "GMT-12"
      },
      "GMT-13": {
        "exemplarCity": "GMT-13"
      },
      "GMT-14": {
        "exemplarCity": "GMT-14"
      },
      "GMT-2": {
        "exemplarCity": "GMT-2"
      },
      "GMT-3": {
        "exemplarCity": "GMT-3"
      },
      "GMT-4": {
        "exemplarCity": "GMT-4"
      },
      "GMT-5": {
        "exemplarCity": "GMT-5"
      },
      "GMT-6": {
        "exemplarCity": "GMT-6"
      },
      "GMT-7": {
        "exemplarCity": "GMT-7"
      },
      "GMT-8": {
        "exemplarCity": "GMT-8"
      },
      "GMT-9": {
        "exemplarCity": "GMT-9"
      },
      "Unknown": {
        "exemplarCity": "مدينة غير معروفة"
      }
    },
    "metazone": {
      "Afghanistan": {
        "long": {
          "standard": "توقيت أفغانستان"
        }
      },
      "Africa_Central": {
        "long": {
          "standard": "توقيت وسط أفريقيا"
        }
      },
      "Africa_Eastern": {
        "long": {
          "standard": "توقيت شرق أفريقيا"
        }
      }
    }
  }

```

```

    }
  },
  "Africa_Southern": {
    "long": {
      "standard": "توقيت جنوب أفريقيا"
    }
  },
  "Africa_Western": {
    "long": {
      "generic": "توقيت غرب أفريقيا",
      "standard": "توقيت غرب أفريقيا الرسمي",
      "daylight": "توقيت غرب أفريقيا الصيفي"
    }
  },
  "Alaska": {
    "long": {
      "generic": "توقيت ألاسكا",
      "standard": "التوقيت الرسمي لألاسكا",
      "daylight": "توقيت ألاسكا الصيفي"
    }
  },
  "Amazon": {
    "long": {
      "generic": "توقيت الأمازون",
      "standard": "توقيت الأمازون الرسمي",
      "daylight": "توقيت الأمازون الصيفي"
    }
  },
  "America_Central": {
    "long": {
      "generic": "التوقيت المركزي لأمريكا الشمالية",
      "standard": "التوقيت الرسمي المركزي لأمريكا الشمالية",
      "daylight": "التوقيت الصيفي المركزي لأمريكا الشمالية"
    }
  },
  "America_Eastern": {
    "long": {
      "generic": "التوقيت الشرقي لأمريكا الشمالية",
      "standard": "التوقيت الرسمي الشرقي لأمريكا الشمالية",
      "daylight": "التوقيت الصيفي الشرقي لأمريكا الشمالية"
    }
  },
  "America_Mountain": {
    "long": {
      "generic": "التوقيت الجبلي لأمريكا الشمالية",
      "standard": "التوقيت الجبلي الرسمي لأمريكا الشمالية",
      "daylight": "التوقيت الجبلي الصيفي لأمريكا الشمالية"
    }
  },
  "America_Pacific": {
    "long": {
      "generic": "توقيت المحيط الهادي",
      "standard": "توقيت المحيط الهادي الرسمي",
      "daylight": "توقيت المحيط الهادي الصيفي"
    }
  },
  "Anadyr": {

```

```

        "long": {
            "generic": "توقيت أنادير",
            "standard": "توقيت أنادير الرسمي",
            "daylight": "التوقيت الصيفي لأنادير"
        }
    },
    "Apia": {
        "long": {
            "generic": "توقيت آبيا",
            "standard": "التوقيت الرسمي لآبيا",
            "daylight": "التوقيت الصيفي لآبيا"
        }
    },
    "Arabian": {
        "long": {
            "generic": "التوقيت العربي",
            "standard": "التوقيت العربي الرسمي",
            "daylight": "التوقيت العربي الصيفي"
        }
    },
    "Argentina": {
        "long": {
            "generic": "توقيت الأرجنتين",
            "standard": "توقيت الأرجنتين الرسمي",
            "daylight": "توقيت الأرجنتين الصيفي"
        }
    },
    "Argentina_Western": {
        "long": {
            "generic": "توقيت غرب الأرجنتين",
            "standard": "توقيت غرب الأرجنتين الرسمي",
            "daylight": "توقيت غرب الأرجنتين الصيفي"
        }
    },
    "Armenia": {
        "long": {
            "generic": "توقيت أرمينيا",
            "standard": "توقيت أرمينيا الرسمي",
            "daylight": "توقيت أرمينيا الصيفي"
        }
    },
    "Atlantic": {
        "long": {
            "generic": "توقيت الأطلسي",
            "standard": "التوقيت الرسمي الأطلسي",
            "daylight": "التوقيت الصيفي الأطلسي"
        }
    },
    "Australia_Central": {
        "long": {
            "generic": "توقيت وسط أستراليا",
            "standard": "توقيت وسط أستراليا الرسمي",
            "daylight": "توقيت وسط أستراليا الصيفي"
        }
    },
    "Australia_CentralWestern": {
        "long": {

```

```

        "generic": "توقيت غرب وسط أستراليا",
        "standard": "توقيت غرب وسط أستراليا الرسمي",
        "daylight": "توقيت غرب وسط أستراليا الصيفي"
    },
},
"Australia_Eastern": {
    "long": {
        "generic": "توقيت شرق أستراليا",
        "standard": "توقيت شرق أستراليا الرسمي",
        "daylight": "توقيت شرق أستراليا الصيفي"
    }
},
"Australia_Western": {
    "long": {
        "generic": "توقيت غرب أستراليا",
        "standard": "توقيت غرب أستراليا الرسمي",
        "daylight": "توقيت غرب أستراليا الصيفي"
    }
},
"Azerbaijan": {
    "long": {
        "generic": "توقيت أذربيجان",
        "standard": "توقيت أذربيجان الرسمي",
        "daylight": "توقيت أذربيجان الصيفي"
    }
},
"Azores": {
    "long": {
        "generic": "توقيت أزورس",
        "standard": "توقيت أزورس الرسمي",
        "daylight": "توقيت أزورس الصيفي"
    }
},
"Bangladesh": {
    "long": {
        "generic": "توقيت بنجلاديش",
        "standard": "توقيت بنجلاديش الرسمي",
        "daylight": "توقيت بنجلاديش الصيفي"
    }
},
"Bhutan": {
    "long": {
        "standard": "توقيت بوتان"
    }
},
"Bolivia": {
    "long": {
        "standard": "توقيت بوليفيا"
    }
},
"Brasilia": {
    "long": {
        "generic": "توقيت برازيليا",
        "standard": "توقيت برازيليا الرسمي",
        "daylight": "توقيت برازيليا الصيفي"
    }
},
},

```

```

"Brunei": {
  "long": {
    "standard": "توقيت بروناي"
  }
},
"Cape_Verde": {
  "long": {
    "generic": "توقيت الرأس الأخضر",
    "standard": "توقيت الرأس الأخضر الرسمي",
    "daylight": "توقيت الرأس الأخضر الصيفي"
  }
},
"Chamorro": {
  "long": {
    "standard": "توقيت تشامورو"
  }
},
"Chatham": {
  "long": {
    "generic": "توقيت تشاتام",
    "standard": "توقيت تشاتام الرسمي",
    "daylight": "توقيت تشاتام الصيفي"
  }
},
"Chile": {
  "long": {
    "generic": "توقيت شيلي",
    "standard": "توقيت شيلي الرسمي",
    "daylight": "توقيت شيلي الصيفي"
  }
},
"China": {
  "long": {
    "generic": "توقيت الصين",
    "standard": "توقيت الصين الرسمي",
    "daylight": "توقيت الصين الصيفي"
  }
},
"Choibalsan": {
  "long": {
    "generic": "توقيت شويبالسان",
    "standard": "توقيت شويبالسان الرسمي",
    "daylight": "التوقيت الصيفي لشويبالسان"
  }
},
"Christmas": {
  "long": {
    "standard": "توقيت جزر الكريسماس"
  }
},
"Cocos": {
  "long": {
    "standard": "توقيت جزر كوكوس"
  }
},
"Colombia": {
  "long": {

```

```

        "generic": "توقيت كولومبيا",
        "standard": "توقيت كولومبيا الرسمي",
        "daylight": "توقيت كولومبيا الصيفي"
    },
},
"Cook": {
    "long": {
        "generic": "توقيت جزر كوك",
        "standard": "توقيت جزر كوك الرسمي",
        "daylight": "توقيت جزر كوك الصيفي"
    }
},
"Cuba": {
    "long": {
        "generic": "توقيت كوبا",
        "standard": "توقيت كوبا الرسمي",
        "daylight": "توقيت كوبا الصيفي"
    }
},
"Davis": {
    "long": {
        "standard": "توقيت دافيز"
    }
},
"DumontDUrville": {
    "long": {
        "standard": "توقيت دي مونت دو روفيل"
    }
},
"East_Timor": {
    "long": {
        "standard": "توقيت تيمور الشرقية"
    }
},
"Easter": {
    "long": {
        "generic": "توقيت جزيرة استر",
        "standard": "توقيت جزيرة استر الرسمي",
        "daylight": "توقيت جزيرة استر الصيفي"
    }
},
"Ecuador": {
    "long": {
        "standard": "توقيت الإكوادور"
    }
},
"Europe_Central": {
    "long": {
        "generic": "توقيت وسط أوروبا",
        "standard": "توقيت وسط أوروبا الرسمي",
        "daylight": "توقيت وسط أوروبا الصيفي"
    }
},
"Europe_Eastern": {
    "long": {
        "generic": "توقيت شرق أوروبا",
        "standard": "توقيت شرق أوروبا الرسمي",

```

```

        "daylight": "توقيت شرق أوروبا الصيفي"
    },
    },
    "Europe_Further_Eastern": {
        "long": {
            "standard": "التوقيت الأوروبي (أكثر شرقًا)"
        }
    },
    "Europe_Western": {
        "long": {
            "generic": "توقيت غرب أوروبا",
            "standard": "توقيت غرب أوروبا الرسمي",
            "daylight": "توقيت غرب أوروبا الصيفي"
        }
    },
    "Falkland": {
        "long": {
            "generic": "توقيت جزر فوكلاند",
            "standard": "توقيت جزر فوكلاند الرسمي",
            "daylight": "توقيت جزر فوكلاند الصيفي"
        }
    },
    "Fiji": {
        "long": {
            "generic": "توقيت فيجي",
            "standard": "توقيت فيجي الرسمي",
            "daylight": "توقيت فيجي الصيفي"
        }
    },
    "French_Guiana": {
        "long": {
            "standard": "توقيت غايانا الفرنسية"
        }
    },
    "French_Southern": {
        "long": {
            "standard": "توقيت المقاطعات الفرنسية الجنوبية والأنتارتيكية"
        }
    },
    "Galapagos": {
        "long": {
            "standard": "توقيت غلاباغوس"
        }
    },
    "Gambier": {
        "long": {
            "standard": "توقيت جامبير"
        }
    },
    "Georgia": {
        "long": {
            "generic": "توقيت جورجيا",
            "standard": "توقيت جورجيا الرسمي",
            "daylight": "توقيت جورجيا الصيفي"
        }
    },
    "Gilbert_Islands": {

```



```

    "long": {
      "standard": "توقيت جزر جيلبرت"
    }
  },
  "GMT": {
    "long": {
      "standard": "توقيت غرينتش"
    }
  },
  "Greenland_Eastern": {
    "long": {
      "generic": "توقيت شرق غرينلاند",
      "standard": "توقيت شرق غرينلاند الرسمي",
      "daylight": "توقيت شرق غرينلاند الصيفي"
    }
  },
  "Greenland_Western": {
    "long": {
      "generic": "توقيت غرب غرينلاند",
      "standard": "توقيت غرب غرينلاند الرسمي",
      "daylight": "توقيت غرب غرينلاند الصيفي"
    }
  },
  "Guam": {
    "long": {
      "standard": "توقيت غوام"
    }
  },
  "Gulf": {
    "long": {
      "standard": "توقيت الخليج"
    }
  },
  "Guyana": {
    "long": {
      "standard": "توقيت غيانا"
    }
  },
  "Hawaii_Aleutian": {
    "long": {
      "generic": "توقيت هاواي ألوتيان",
      "standard": "توقيت هاواي ألوتيان الرسمي",
      "daylight": "توقيت هاواي ألوتيان الصيفي"
    }
  },
  "Hong_Kong": {
    "long": {
      "generic": "توقيت هونغ كونغ",
      "standard": "توقيت هونغ كونغ الرسمي",
      "daylight": "توقيت هونغ كونغ الصيفي"
    }
  },
  "Hovd": {
    "long": {
      "generic": "توقيت هوفد",
      "standard": "توقيت هوفد الرسمي",
      "daylight": "توقيت هوفد الصيفي"
    }
  }
}

```

```

    }
  },
  "India": {
    "long": {
      "standard": "توقيت الهند"
    }
  },
  "Indian_Ocean": {
    "long": {
      "standard": "توقيت المحيط الهندي"
    }
  },
  "Indochina": {
    "long": {
      "standard": "توقيت الهند الصينية"
    }
  },
  "Indonesia_Central": {
    "long": {
      "standard": "توقيت وسط إندونيسيا"
    }
  },
  "Indonesia_Eastern": {
    "long": {
      "standard": "توقيت شرق إندونيسيا"
    }
  },
  "Indonesia_Western": {
    "long": {
      "standard": "توقيت غرب إندونيسيا"
    }
  },
  "Iran": {
    "long": {
      "generic": "توقيت إيران",
      "standard": "توقيت إيران الرسمي",
      "daylight": "توقيت إيران الصيفي"
    }
  },
  "Irkutsk": {
    "long": {
      "generic": "توقيت إركوتسك",
      "standard": "توقيت إركوتسك الرسمي",
      "daylight": "توقيت إركوتسك الصيفي"
    }
  },
  "Israel": {
    "long": {
      "generic": "توقيت إسرائيل",
      "standard": "توقيت إسرائيل الرسمي",
      "daylight": "توقيت إسرائيل الصيفي"
    }
  },
  "Japan": {
    "long": {
      "generic": "توقيت اليابان",
      "standard": "توقيت اليابان الرسمي",

```

```

        "daylight": "توقيت اليابان الصيفي"
    },
    },
    "Kamchatka": {
        "long": {
            "generic": "توقيت كامشاتكا",
            "standard": "توقيت بيتروبافلوفسك-كامتشاتسكي",
            "daylight": "توقيت بيتروبافلوفسك-كامتشاتسكي الصيفي"
        }
    },
    "Kazakhstan_Eastern": {
        "long": {
            "standard": "توقيت شرق كازاخستان"
        }
    },
    "Kazakhstan_Western": {
        "long": {
            "standard": "توقيت غرب كازاخستان"
        }
    },
    "Korea": {
        "long": {
            "generic": "توقيت كوريا",
            "standard": "توقيت كوريا الرسمي",
            "daylight": "توقيت كوريا الصيفي"
        }
    },
    "Kosrae": {
        "long": {
            "standard": "توقيت كوسرا"
        }
    },
    "Krasnoyarsk": {
        "long": {
            "generic": "توقيت كراسنويارسك",
            "standard": "توقيت كراسنويارسك الرسمي",
            "daylight": "التوقيت الصيفي لكراسنويارسك"
        }
    },
    "Kyrgystan": {
        "long": {
            "standard": "توقيت قرغيزستان"
        }
    },
    "Line_Islands": {
        "long": {
            "standard": "توقيت جزر لايين"
        }
    },
    "Lord_Howe": {
        "long": {
            "generic": "توقيت لورد هاو",
            "standard": "توقيت لورد هاو الرسمي",
            "daylight": "التوقيت الصيفي للورد هاو"
        }
    },
    "Macquarie": {

```

```

        "long": {
            "standard": "توقيت ماكواري"
        }
    },
    "Magadan": {
        "long": {
            "generic": "توقيت ماغادان",
            "standard": "توقيت ماغادان الرسمي",
            "daylight": "توقيت ماغادان الصيفي"
        }
    },
    "Malaysia": {
        "long": {
            "standard": "توقيت ماليزيا"
        }
    },
    "Maldives": {
        "long": {
            "standard": "توقيت المالديف"
        }
    },
    "Marquesas": {
        "long": {
            "standard": "توقيت ماركيساس"
        }
    },
    "Marshall_Islands": {
        "long": {
            "standard": "توقيت جزر مارشال"
        }
    },
    "Mauritius": {
        "long": {
            "generic": "توقيت موريشيوس",
            "standard": "توقيت موريشيوس الرسمي",
            "daylight": "توقيت موريشيوس الصيفي"
        }
    },
    "Mawson": {
        "long": {
            "standard": "توقيت ماوسون"
        }
    },
    "Mexico_Northwest": {
        "long": {
            "generic": "توقيت شمال غرب المكسيك",
            "standard": "التوقيت الرسمي لشمال غرب المكسيك",
            "daylight": "التوقيت الصيفي لشمال غرب المكسيك"
        }
    },
    "Mexico_Pacific": {
        "long": {
            "generic": "توقيت المحيط الهادي للمكسيك",
            "standard": "توقيت المحيط الهادي الرسمي للمكسيك",
            "daylight": "توقيت المحيط الهادي الصيفي للمكسيك"
        }
    },
    },

```

```

"Mongolia": {
  "long": {
    "generic": "توقيت أولان باتور",
    "standard": "توقيت أولان باتور الرسمي",
    "daylight": "توقيت أولان باتور الصيفي"
  }
},
"Moscow": {
  "long": {
    "generic": "توقيت موسكو",
    "standard": "توقيت موسكو الرسمي",
    "daylight": "توقيت موسكو الصيفي"
  }
},
"Myanmar": {
  "long": {
    "standard": "توقيت ميانمار"
  }
},
"Nauru": {
  "long": {
    "standard": "توقيت ناورو"
  }
},
"Nepal": {
  "long": {
    "standard": "توقيت نيبال"
  }
},
"New_Caledonia": {
  "long": {
    "generic": "توقيت كاليدونيا الجديدة",
    "standard": "توقيت كاليدونيا الجديدة الرسمي",
    "daylight": "توقيت كاليدونيا الجديدة الصيفي"
  }
},
"New_Zealand": {
  "long": {
    "generic": "توقيت نيوزيلندا",
    "standard": "توقيت نيوزيلندا الرسمي",
    "daylight": "توقيت نيوزيلندا الصيفي"
  }
},
"Newfoundland": {
  "long": {
    "generic": "توقيت نيوفاوندلاند",
    "standard": "توقيت نيوفاوندلاند الرسمي",
    "daylight": "توقيت نيوفاوندلاند الصيفي"
  }
},
"Niue": {
  "long": {
    "standard": "توقيت نيوي"
  }
},
"Norfolk": {
  "long": {

```

```

        "standard": "توقيت جزيرة نورفولك"
    },
    },
    "Noronha": {
        "long": {
            "generic": "توقيت فيرناندو دي نورونها",
            "standard": "توقيت فرناندو دي نورونها الرسمي",
            "daylight": "توقيت فرناندو دي نورونها الصيفي"
        }
    },
    "North_Mariana": {
        "long": {
            "standard": "توقيت جزر ماريانا الشمالية"
        }
    },
    "Novosibirsk": {
        "long": {
            "generic": "توقيت نوفوسيبيرسك",
            "standard": "توقيت نوفوسيبيرسك الرسمي",
            "daylight": "توقيت نوفوسيبيرسك الصيفي"
        }
    },
    "Omsk": {
        "long": {
            "generic": "توقيت أومسك",
            "standard": "توقيت أومسك الرسمي",
            "daylight": "توقيت أومسك الصيفي"
        }
    },
    "Pakistan": {
        "long": {
            "generic": "توقيت باكستان",
            "standard": "توقيت باكستان الرسمي",
            "daylight": "توقيت باكستان الصيفي"
        }
    },
    "Palau": {
        "long": {
            "standard": "توقيت بالاو"
        }
    },
    "Papua_New_Guinea": {
        "long": {
            "standard": "توقيت بابوا غينيا الجديدة"
        }
    },
    "Paraguay": {
        "long": {
            "generic": "توقيت باراغواي",
            "standard": "توقيت باراغواي الرسمي",
            "daylight": "توقيت باراغواي الصيفي"
        }
    },
    "Peru": {
        "long": {
            "generic": "توقيت بيرو",
            "standard": "توقيت بيرو الرسمي",

```

```

        "daylight": "توقيت بيرو الصيفي"
    },
    },
    "Philippines": {
        "long": {
            "generic": "توقيت الفلبين",
            "standard": "توقيت الفلبين الرسمي",
            "daylight": "توقيت الفلبين الصيفي"
        }
    },
    "Phoenix_Islands": {
        "long": {
            "standard": "توقيت جزر فينكس"
        }
    },
    "Pierre_Miquelon": {
        "long": {
            "generic": "توقيت سانت بيير وميكلون",
            "standard": "توقيت سانت بيير وميكلون الرسمي",
            "daylight": "توقيت سانت بيير وميكلون الصيفي"
        }
    },
    "Pitcairn": {
        "long": {
            "standard": "توقيت بيتكيرن"
        }
    },
    "Ponape": {
        "long": {
            "standard": "توقيت بونابي"
        }
    },
    "Pyongyang": {
        "long": {
            "standard": "توقيت بيونغ يانغ"
        }
    },
    "Reunion": {
        "long": {
            "standard": "توقيت ريونيون"
        }
    },
    "Rothera": {
        "long": {
            "standard": "توقيت روثيرا"
        }
    },
    "Sakhalin": {
        "long": {
            "generic": "توقيت ساخالين",
            "standard": "توقيت ساخالين الرسمي",
            "daylight": "توقيت ساخالين الصيفي"
        }
    },
    "Samara": {
        "long": {
            "generic": "توقيت سامارا",

```

```

        "standard": "توقيت سمارا",
        "daylight": "توقيت سمارا الصيفي"
    },
    "Samoa": {
        "long": {
            "generic": "توقيت ساموا",
            "standard": "توقيت ساموا الرسمي",
            "daylight": "توقيت ساموا الصيفي"
        }
    },
    "Seychelles": {
        "long": {
            "standard": "توقيت سيشل"
        }
    },
    "Singapore": {
        "long": {
            "standard": "توقيت سنغافورة"
        }
    },
    "Solomon": {
        "long": {
            "standard": "توقيت جزر سليمان"
        }
    },
    "South_Georgia": {
        "long": {
            "standard": "توقيت جنوب جورجيا"
        }
    },
    "Suriname": {
        "long": {
            "standard": "توقيت سورينام"
        }
    },
    "Syowa": {
        "long": {
            "standard": "توقيت سايووا"
        }
    },
    "Tahiti": {
        "long": {
            "standard": "توقيت تاهيتي"
        }
    },
    "Taipei": {
        "long": {
            "generic": "توقيت تايبيه",
            "standard": "توقيت تايبيه الرسمي",
            "daylight": "توقيت تايبيه الصيفي"
        }
    },
    "Tajikistan": {
        "long": {
            "standard": "توقيت طاجكستان"
        }
    }

```



```

    },
    "Tokelau": {
      "long": {
        "standard": "توقيت توكيلاو"
      }
    },
    "Tonga": {
      "long": {
        "generic": "توقيت تونغا",
        "standard": "توقيت تونغا الرسمي",
        "daylight": "توقيت تونغا الصيفي"
      }
    },
    "Truk": {
      "long": {
        "standard": "توقيت شوك"
      }
    },
    "Turkmenistan": {
      "long": {
        "generic": "توقيت تركمانستان",
        "standard": "توقيت تركمانستان الرسمي",
        "daylight": "توقيت تركمانستان الصيفي"
      }
    },
    "Tuvalu": {
      "long": {
        "standard": "توقيت توفالو"
      }
    },
    "Uruguay": {
      "long": {
        "generic": "توقيت أورغواي",
        "standard": "توقيت أورغواي الرسمي",
        "daylight": "توقيت أورغواي الصيفي"
      }
    },
    "Uzbekistan": {
      "long": {
        "generic": "توقيت أوزبكستان",
        "standard": "توقيت أوزبكستان الرسمي",
        "daylight": "توقيت أوزبكستان الصيفي"
      }
    },
    "Vanuatu": {
      "long": {
        "generic": "توقيت فانواتو",
        "standard": "توقيت فانواتو الرسمي",
        "daylight": "توقيت فانواتو الصيفي"
      }
    },
    "Venezuela": {
      "long": {
        "standard": "توقيت فنزويلا"
      }
    },
    "Vladivostok": {

```

```

        "long": {
          "generic": "توقيت فلاديفوستوك",
          "standard": "توقيت فلاديفوستوك الرسمي",
          "daylight": "توقيت فلاديفوستوك الصيفي"
        }
      },
      "Volgograd": {
        "long": {
          "generic": "توقيت فولغوغراد",
          "standard": "توقيت فولغوغراد الرسمي",
          "daylight": "توقيت فولغوغراد الصيفي"
        }
      },
      "Vostok": {
        "long": {
          "standard": "توقيت فوستوك"
        }
      },
      "Wake": {
        "long": {
          "standard": "توقيت جزيرة ويك"
        }
      },
      "Wallis": {
        "long": {
          "standard": "توقيت واليس و فوتونا"
        }
      },
      "Yakutsk": {
        "long": {
          "generic": "توقيت ياكوتسك",
          "standard": "توقيت ياكوتسك الرسمي",
          "daylight": "توقيت ياكوتسك الصيفي"
        }
      },
      "Yekaterinburg": {
        "long": {
          "generic": "توقيت يكاترينبورغ",
          "standard": "توقيت يكاترينبورغ الرسمي",
          "daylight": "توقيت يكاترينبورغ الصيفي"
        }
      }
    }
  }
}

```

TIMEZONENAMES.JSX

```

{
  "main";
  {
    "ar";
    {

```

```

"identity";
{
  "version";
  {
    "_number";
    "$Revision: 12879 $",
    "_cldrVersion";
    "30.0.3";
  }
  "language";
  "ar";
}
"dates";
{
  "timeZoneNames";
  {
    "hourFormat";
    "+HH:mm;-HH:mm",
    "gmtFormat";
    "0{جرينتش",
    "gmtZeroFormat";
    "جرينتش",
    "regionFormat";
    "0{توقيت",
    "regionFormat-type-daylight";
    "الصيفي } 0{توقيت",
    "regionFormat-type-standard";
    "الرسمي } 0{توقيت",
    "fallbackFormat";
    "{1} ({0})",
    "zone";
    {
      "America";
      {
        "Adak";
        {
          "exemplarCity";
          "أداك";
        }
        "Anchorage";
        {
          "exemplarCity";
          "أنشوراج";
        }
        "Anguilla";
        {
          "exemplarCity";
          "أنغيلا";
        }
        "Antigua";
        {
          "exemplarCity";
          "أنتيغوا";
        }
        "Araguaina";
        {
          "exemplarCity";

```

```

        "أروجوانيا";
    }
    "Argentina";
    {
        "Rio_Gallegos";
        {
            "exemplarCity";
            "ريو جالييوس";
        }
        "San_Juan";
        {
            "exemplarCity";
            "سان خوان";
        }
        "Ushuaia";
        {
            "exemplarCity";
            "أشوا";
        }
        "La_Rioja";
        {
            "exemplarCity";
            "لا ريوجا";
        }
        "San_Luis";
        {
            "exemplarCity";
            "سان لويس";
        }
        "Salta";
        {
            "exemplarCity";
            "سالطا";
        }
        "Tucuman";
        {
            "exemplarCity";
            "تاكمان";
        }
    }
    "Aruba";
    {
        "exemplarCity";
        "أروبا";
    }
    "Asuncion";
    {
        "exemplarCity";
        "أسونسيون";
    }
    "Bahia";
    {
        "exemplarCity";
        "باهيا";
    }
    "Bahia_Banderas";
    {

```

```

        "exemplarCity";
        "باهيا بانديراس";
    }
    "Barbados";
    {
        "exemplarCity";
        "بربادوس";
    }
    "Belem";
    {
        "exemplarCity";
        "بلم";
    }
    "Belize";
    {
        "exemplarCity";
        "بليز";
    }
    "Blanc-Sablon";
    {
        "exemplarCity";
        "بلانك-سابلون";
    }
    "Boa_Vista";
    {
        "exemplarCity";
        "باو فيستا";
    }
    "Bogota";
    {
        "exemplarCity";
        "بوغوتا";
    }
    "Boise";
    {
        "exemplarCity";
        "بويس";
    }
    "Buenos_Aires";
    {
        "exemplarCity";
        "بوينوس آيرس";
    }
    "Cambridge_Bay";
    {
        "exemplarCity";
        "كامبرديج باي";
    }
    "Campo_Grande";
    {
        "exemplarCity";
        "كومبو جراند";
    }
    "Cancun";
    {
        "exemplarCity";
        "كانكون";
    }

```

```
}
"Caracas";
{
  "exemplarCity";
  "کاراکاس";
}
"Catamarca";
{
  "exemplarCity";
  "کاتامارکا";
}
"Cayenne";
{
  "exemplarCity";
  "کایین";
}
"Cayman";
{
  "exemplarCity";
  "کیمان";
}
"Chicago";
{
  "exemplarCity";
  "شیکاگو";
}
"Chihuahua";
{
  "exemplarCity";
  "تشیواوا";
}
"Coral_Harbour";
{
  "exemplarCity";
  "کورانال هاربر";
}
"Cordoba";
{
  "exemplarCity";
  "کوردوبا";
}
"Costa_Rica";
{
  "exemplarCity";
  "کوستاریکا";
}
"Creston";
{
  "exemplarCity";
  "کریستون";
}
"Cuiaba";
{
  "exemplarCity";
  "کیابا";
}
"Curacao";
```

```
{
  "exemplarCity";
  "كوراكاو";
}
"Danmarkshavn";
{
  "exemplarCity";
  "دانمرك شافن";
}
"Dawson";
{
  "exemplarCity";
  "داوسان";
}
"Dawson_Creek";
{
  "exemplarCity";
  "داوسن كريك";
}
"Denver";
{
  "exemplarCity";
  "دنفر";
}
"Detroit";
{
  "exemplarCity";
  "ديترويت";
}
"Dominica";
{
  "exemplarCity";
  "دومينيكا";
}
"Edmonton";
{
  "exemplarCity";
  "ايدمونتون";
}
"Eirunepe";
{
  "exemplarCity";
  "ايرونبي";
}
"El_Salvador";
{
  "exemplarCity";
  "السلفادور";
}
"Fort_Nelson";
{
  "exemplarCity";
  "فورت نيلسون";
}
"Fortaleza";
{
  "exemplarCity";
```

```

        "فورتاليزا";
    }
    "Glace_Bay";
    {
        "exemplarCity";
        "جلاس باي";
    }
    "Godthab";
    {
        "exemplarCity";
        "غودثاب";
    }
    "Goose_Bay";
    {
        "exemplarCity";
        "جوس باي";
    }
    "Grand_Turk";
    {
        "exemplarCity";
        "غراند ترك";
    }
    "Grenada";
    {
        "exemplarCity";
        "جرينادا";
    }
    "Guadeloupe";
    {
        "exemplarCity";
        "غوادلوب";
    }
    "Guatemala";
    {
        "exemplarCity";
        "غواتيمالا";
    }
    "Guayaquil";
    {
        "exemplarCity";
        "غواياكويل";
    }
    "Guyana";
    {
        "exemplarCity";
        "غيانا";
    }
    "Halifax";
    {
        "exemplarCity";
        "هاليفاكس";
    }
    "Havana";
    {
        "exemplarCity";
        "هافانا";
    }
}

```



```

"Hermosillo";
{
  "exemplarCity";
  "هيرموسيلو";
}
"Indiana";
{
  "Vincennes";
  {
    "exemplarCity";
    "فينسينس";
  }
  "Petersburg";
  {
    "exemplarCity";
    "بيتربيرغ";
  }
  "Tell_City";
  {
    "exemplarCity";
    "مدينة تل، إنديانا";
  }
  "Knox";
  {
    "exemplarCity";
    "كونكس";
  }
  "Winamac";
  {
    "exemplarCity";
    "ويناماك";
  }
  "Marengo";
  {
    "exemplarCity";
    "مارنجو";
  }
  "Vevay";
  {
    "exemplarCity";
    "فيفاي";
  }
}
"Indianapolis";
{
  "exemplarCity";
  "إنديانا بوليس";
}
"Inuvik";
{
  "exemplarCity";
  "اينوفيك";
}
"Iqaluit";
{
  "exemplarCity";
  "اكويلت";
}

```

```
}
"Jamaica";
{
  "exemplarCity";
  "جاماىكا";
}
"Jujuy";
{
  "exemplarCity";
  "جوجو";
}
"Juneau";
{
  "exemplarCity";
  "جونى";
}
"Kentucky";
{
  "Monticello";
  {
    "exemplarCity";
    "مونتي سيلو";
  }
}
"Kralendijk";
{
  "exemplarCity";
  "كرالندىك";
}
"La_Paz";
{
  "exemplarCity";
  "لا باز";
}
"Lima";
{
  "exemplarCity";
  "ليما";
}
"Los_Angeles";
{
  "exemplarCity";
  "لوس انجلوس";
}
"Louisville";
{
  "exemplarCity";
  "لويس فيل";
}
"Lower_Princes";
{
  "exemplarCity";
  "حي الأمير السفلي";
}
"Maceio";
{
  "exemplarCity";
```

```

        "ماشيو";
    }
    "Managua";
    {
        "exemplarCity";
        "ماناغوا";
    }
    "Manaus";
    {
        "exemplarCity";
        "ماناوس";
    }
    "Marigot";
    {
        "exemplarCity";
        "ماريغوت";
    }
    "Martinique";
    {
        "exemplarCity";
        "المارتينيك";
    }
    "Matamoros";
    {
        "exemplarCity";
        "ماتاموروس";
    }
    "Mazatlan";
    {
        "exemplarCity";
        "مازاتلان";
    }
    "Mendoza";
    {
        "exemplarCity";
        "ميندوزا";
    }
    "Menominee";
    {
        "exemplarCity";
        "مينوميني";
    }
    "Merida";
    {
        "exemplarCity";
        "ميريدا";
    }
    "Metlakatla";
    {
        "exemplarCity";
        "ميتلاكاتلا";
    }
    "Mexico_City";
    {
        "exemplarCity";
        "مدينة المكسيك";
    }
}

```

```

"Miquelon";
{
  "exemplarCity";
  "ميكلون";
}
"Moncton";
{
  "exemplarCity";
  "وينكتون";
}
"Monterrey";
{
  "exemplarCity";
  "مونتيري";
}
"Montevideo";
{
  "exemplarCity";
  "مونتيفيديو";
}
"Montserrat";
{
  "exemplarCity";
  "مونتسيرات";
}
"Nassau";
{
  "exemplarCity";
  "ناسو";
}
"New_York";
{
  "exemplarCity";
  "نيويورك";
}
"Nipigon";
{
  "exemplarCity";
  "نيبيجون";
}
"Nome";
{
  "exemplarCity";
  "نوم";
}
"Noronha";
{
  "exemplarCity";
  "نوروناه";
}
"North_Dakota";
{
  "Beulah";
  {
    "exemplarCity";
    "بيولا، داکوتا الشمالية";
  }
}

```

```

        "New_Salem";
        {
            "exemplarCity";
            "نيو ساليم";
        }
        "Center";
        {
            "exemplarCity";
            "سنتر";
        }
    }
    "Ojinaga";
    {
        "exemplarCity";
        "أوجيناغا";
    }
    "Panama";
    {
        "exemplarCity";
        "بنما";
    }
    "Pangnirtung";
    {
        "exemplarCity";
        "بانجينتینگ";
    }
    "Paramaribo";
    {
        "exemplarCity";
        "باراماريبو";
    }
    "Phoenix";
    {
        "exemplarCity";
        "فينكس";
    }
    "Port-au-Prince";
    {
        "exemplarCity";
        "بورت أو برنس";
    }
    "Port_of_Spain";
    {
        "exemplarCity";
        "بورت أوف سبين";
    }
    "Porto_Velho";
    {
        "exemplarCity";
        "بورتو فيلو";
    }
    "Puerto_Rico";
    {
        "exemplarCity";
        "بورتوريكو";
    }
    "Rainy_River";

```

```
{
    "exemplarCity";
    "راني ريفر";
}
"Rankin_Inlet";
{
    "exemplarCity";
    "رانكن انلت";
}
"Recife";
{
    "exemplarCity";
    "ريسيف";
}
"Regina";
{
    "exemplarCity";
    "ريجينا";
}
"Resolute";
{
    "exemplarCity";
    "ريزولوت";
}
"Rio_Branco";
{
    "exemplarCity";
    "ريوبرانكو";
}
"Santa_Isabel";
{
    "exemplarCity";
    "سانتا إيزابيل";
}
"Santarem";
{
    "exemplarCity";
    "سانتاريم";
}
"Santiago";
{
    "exemplarCity";
    "سانتياغو";
}
"Santo_Domingo";
{
    "exemplarCity";
    "سانتو دومينغو";
}
"Sao_Paulo";
{
    "exemplarCity";
    "ساو باولو";
}
"Scoresbysund";
{
    "exemplarCity";
```

```
        "سکورسبیسند";
    }
    "Sitka";
    {
        "exemplarCity";
        "سیتکا";
    }
    "St_Barthelemy";
    {
        "exemplarCity";
        "سانت بارتیلیمی";
    }
    "St_Johns";
    {
        "exemplarCity";
        "سانت جونس";
    }
    "St_Kitts";
    {
        "exemplarCity";
        "سانت کیتس";
    }
    "St_Lucia";
    {
        "exemplarCity";
        "سانت لوشیا";
    }
    "St_Thomas";
    {
        "exemplarCity";
        "سانت توماس";
    }
    "St_Vincent";
    {
        "exemplarCity";
        "سانت فنسنت";
    }
    "Swift_Current";
    {
        "exemplarCity";
        "سوفت کارنت";
    }
    "Tegucigalpa";
    {
        "exemplarCity";
        "تیغوسیغالبا";
    }
    "Thule";
    {
        "exemplarCity";
        "ٹیل";
    }
    "Thunder_Bay";
    {
        "exemplarCity";
        "ٹندر بای";
    }
}
```

```
"Tijuana";
{
  "exemplarCity";
  "تيخوانا";
}
"Toronto";
{
  "exemplarCity";
  "تورونتو";
}
"Tortola";
{
  "exemplarCity";
  "تورتولا";
}
"Vancouver";
{
  "exemplarCity";
  "فانكوفر";
}
"Whitehorse";
{
  "exemplarCity";
  "وايت هورس";
}
"Winnipeg";
{
  "exemplarCity";
  "وينيبيج";
}
"Yakutat";
{
  "exemplarCity";
  "ياكوتات";
}
"Yellowknife";
{
  "exemplarCity";
  "يلونيف";
}
}
"Atlantic";
{
  "Azores";
  {
    "exemplarCity";
    "أزورس";
  }
  "Bermuda";
  {
    "exemplarCity";
    "برمودا";
  }
  "Canary";
  {
    "exemplarCity";
    "كناري";
  }
}
```



```

    }
    "Cape_Verde";
    {
        "exemplarCity";
        "الرأس الأخضر";
    }
    "Faeroe";
    {
        "exemplarCity";
        "فارو";
    }
    "Madeira";
    {
        "exemplarCity";
        "ماديرا";
    }
    "Reykjavik";
    {
        "exemplarCity";
        "ريكيافيك";
    }
    "South_Georgia";
    {
        "exemplarCity";
        "جورجيا الجنوبية";
    }
    "St_Helena";
    {
        "exemplarCity";
        "سانت هيلينا";
    }
    "Stanley";
    {
        "exemplarCity";
        "استانلي";
    }
}
"Europe";
{
    "Amsterdam";
    {
        "exemplarCity";
        "أمستردام";
    }
    "Andorra";
    {
        "exemplarCity";
        "أندورا";
    }
    "Astrakhan";
    {
        "exemplarCity";
        "أستراخان";
    }
    "Athens";
    {
        "exemplarCity";
    }
}

```

```

        "أثينا";
    }
    "Belgrade";
    {
        "exemplarCity";
        "بلغراد";
    }
    "Berlin";
    {
        "exemplarCity";
        "برلين";
    }
    "Bratislava";
    {
        "exemplarCity";
        "براتيسلافا";
    }
    "Brussels";
    {
        "exemplarCity";
        "بروكسل";
    }
    "Bucharest";
    {
        "exemplarCity";
        "بوخارست";
    }
    "Budapest";
    {
        "exemplarCity";
        "بودابست";
    }
    "Busingen";
    {
        "exemplarCity";
        "بوسنغن";
    }
    "Chisinau";
    {
        "exemplarCity";
        "تشيسيناو";
    }
    "Copenhagen";
    {
        "exemplarCity";
        "كوبنهاغن";
    }
    "Dublin";
    {
        "long";
        {
            "daylight";
            "توقيت أيرلندا الرسمي";
        }
        "exemplarCity";
        "دبلن";
    }
}

```

```
"Gibraltar";
{
  "exemplarCity";
  "جبل طارق";
}
"Guernsey";
{
  "exemplarCity";
  "غيرنسي";
}
"Helsinki";
{
  "exemplarCity";
  "هلسنكي";
}
"Isle_of_Man";
{
  "exemplarCity";
  "جزيرة مان";
}
"Istanbul";
{
  "exemplarCity";
  "إسطنبول";
}
"Jersey";
{
  "exemplarCity";
  "جيرسي";
}
"Kaliningrad";
{
  "exemplarCity";
  "كالينجراد";
}
"Kiev";
{
  "exemplarCity";
  "كييف";
}
"Kirov";
{
  "exemplarCity";
  "كيروف";
}
"Lisbon";
{
  "exemplarCity";
  "لشبونة";
}
"Ljubljana";
{
  "exemplarCity";
  "ليوبليانا";
}
"London";
{
```

```
        "long";
        {
            "daylight";
            "توقيت بريطانيا الصيفي";
        }
        "exemplarCity";
        "لندن";
    }
    "Luxembourg";
    {
        "exemplarCity";
        "لوكسمبورغ";
    }
    "Madrid";
    {
        "exemplarCity";
        "مدريد";
    }
    "Malta";
    {
        "exemplarCity";
        "مالطة";
    }
    "Mariehamn";
    {
        "exemplarCity";
        "ماريهامن";
    }
    "Minsk";
    {
        "exemplarCity";
        "مينسك";
    }
    "Monaco";
    {
        "exemplarCity";
        "موناكو";
    }
    "Moscow";
    {
        "exemplarCity";
        "موسكو";
    }
    "Oslo";
    {
        "exemplarCity";
        "أوسلو";
    }
    "Paris";
    {
        "exemplarCity";
        "باريس";
    }
    "Podgorica";
    {
        "exemplarCity";
        "بودغوريكا";
    }
}
```

```
}
"Prague";
{
  "exemplarCity";
  "براغ";
}
"Riga";
{
  "exemplarCity";
  "ريغا";
}
"Rome";
{
  "exemplarCity";
  "روما";
}
"Samara";
{
  "exemplarCity";
  "سمراء";
}
"San_Marino";
{
  "exemplarCity";
  "سان مارينو";
}
"Sarajevo";
{
  "exemplarCity";
  "سراييفو";
}
"Simferopol";
{
  "exemplarCity";
  "سيمفروبول";
}
"Skopje";
{
  "exemplarCity";
  "سكوبي";
}
"Sofia";
{
  "exemplarCity";
  "صوفيا";
}
"Stockholm";
{
  "exemplarCity";
  "ستوكهولم";
}
"Tallinn";
{
  "exemplarCity";
  "تالين";
}
"Tirane";
```

```
{
    "exemplarCity";
    "تيرانا";
}
"Ulyanovsk";
{
    "exemplarCity";
    "أوليانوفسك";
}
"Uzhgorod";
{
    "exemplarCity";
    "أوزجروود";
}
"Vaduz";
{
    "exemplarCity";
    "فادوز";
}
"Vatican";
{
    "exemplarCity";
    "الفاتيكان";
}
"Vienna";
{
    "exemplarCity";
    "فيينا";
}
"Vilnius";
{
    "exemplarCity";
    "فيلنيوس";
}
"Volgograd";
{
    "exemplarCity";
    "فولجوراد";
}
"Warsaw";
{
    "exemplarCity";
    "وارسو";
}
"Zagreb";
{
    "exemplarCity";
    "زغرب";
}
"Zaporozhye";
{
    "exemplarCity";
    "زابوروزي";
}
"Zurich";
{
    "exemplarCity";
```

```

        "زيورخ";
    }
}
"Africa";
{
    "Abidjan";
    {
        "exemplarCity";
        "أبيدجان";
    }
    "Accra";
    {
        "exemplarCity";
        "أكرا";
    }
    "Addis_Ababa";
    {
        "exemplarCity";
        "أديس أبابا";
    }
    "Algiers";
    {
        "exemplarCity";
        "الجزائر";
    }
    "Asmera";
    {
        "exemplarCity";
        "أسمرّة";
    }
    "Bamako";
    {
        "exemplarCity";
        "باماكو";
    }
    "Bangui";
    {
        "exemplarCity";
        "بانغوي";
    }
    "Banjul";
    {
        "exemplarCity";
        "بانجول";
    }
    "Bissau";
    {
        "exemplarCity";
        "بيساو";
    }
    "Blantyre";
    {
        "exemplarCity";
        "بلانتاير";
    }
    "Brazzaville";
    {

```

```
        "exemplarCity";
        "برازافيل";
    }
    "Bujumbura";
    {
        "exemplarCity";
        "بوجومبورا";
    }
    "Cairo";
    {
        "exemplarCity";
        "القاهرة";
    }
    "Casablanca";
    {
        "exemplarCity";
        "الدار البيضاء";
    }
    "Ceuta";
    {
        "exemplarCity";
        "سيتا";
    }
    "Conakry";
    {
        "exemplarCity";
        "كوناكري";
    }
    "Dakar";
    {
        "exemplarCity";
        "داكار";
    }
    "Dar_es_Salaam";
    {
        "exemplarCity";
        "دار السلام";
    }
    "Djibouti";
    {
        "exemplarCity";
        "جيبوتي";
    }
    "Douala";
    {
        "exemplarCity";
        "دوالا";
    }
    "El_Aaiun";
    {
        "exemplarCity";
        "العيون";
    }
    "Freetown";
    {
        "exemplarCity";
        "فري تاون";
    }
```



```
}
"Gaborone";
{
  "exemplarCity";
  "غابورون";
}
"Harare";
{
  "exemplarCity";
  "هراري";
}
"Johannesburg";
{
  "exemplarCity";
  "جوهانسبرغ";
}
"Juba";
{
  "exemplarCity";
  "جوبا";
}
"Kampala";
{
  "exemplarCity";
  "كامبالا";
}
"Khartoum";
{
  "exemplarCity";
  "الخرطوم";
}
"Kigali";
{
  "exemplarCity";
  "كيغالي";
}
"Kinshasa";
{
  "exemplarCity";
  "كينشاسا";
}
"Lagos";
{
  "exemplarCity";
  "لاغوس";
}
"Libreville";
{
  "exemplarCity";
  "ليبرفيل";
}
"Lome";
{
  "exemplarCity";
  "لومي";
}
"Luanda";
```

```
{
  "exemplarCity";
  "لواندا";
}
"Lubumbashi";
{
  "exemplarCity";
  "لومبباشا";
}
"Lusaka";
{
  "exemplarCity";
  "لوساكا";
}
"Malabo";
{
  "exemplarCity";
  "مالابو";
}
"Maputo";
{
  "exemplarCity";
  "مابوتو";
}
"Maseru";
{
  "exemplarCity";
  "ماسيرو";
}
"Mbabane";
{
  "exemplarCity";
  "مباباني";
}
"Mogadishu";
{
  "exemplarCity";
  "مقديشيو";
}
"Monrovia";
{
  "exemplarCity";
  "مونروفيا";
}
"Nairobi";
{
  "exemplarCity";
  "نيروبي";
}
"Ndjamena";
{
  "exemplarCity";
  "نجامينا";
}
"Niamey";
{
  "exemplarCity";
```

```

        "نيامي";
    }
    "Nouakchott";
    {
        "exemplarCity";
        "نواكشوط";
    }
    "Ouagadougou";
    {
        "exemplarCity";
        "واغادوغو";
    }
    "Porto-Novo";
    {
        "exemplarCity";
        "بورتو نوفو";
    }
    "Sao_Tome";
    {
        "exemplarCity";
        "ساو تومي";
    }
    "Tripoli";
    {
        "exemplarCity";
        "طرابلس";
    }
    "Tunis";
    {
        "exemplarCity";
        "تونس";
    }
    "Windhoek";
    {
        "exemplarCity";
        "ويندهوك";
    }
}
"Asia";
{
    "Aden";
    {
        "exemplarCity";
        "عدن";
    }
    "Almaty";
    {
        "exemplarCity";
        "ألماتي";
    }
    "Amman";
    {
        "exemplarCity";
        "عمان";
    }
    "Anadyr";
    {

```

```
        "exemplarCity";
        "أندير";
    }
    "Aqtau";
    {
        "exemplarCity";
        "أكتاو";
    }
    "Aqtobe";
    {
        "exemplarCity";
        "أکتوب";
    }
    "Ashgabat";
    {
        "exemplarCity";
        "عشق آباد";
    }
    "Baghdad";
    {
        "exemplarCity";
        "بغداد";
    }
    "Bahrain";
    {
        "exemplarCity";
        "البحرين";
    }
    "Baku";
    {
        "exemplarCity";
        "باکو";
    }
    "Bangkok";
    {
        "exemplarCity";
        "بانكوك";
    }
    "Barnaul";
    {
        "exemplarCity";
        "بارناول";
    }
    "Beirut";
    {
        "exemplarCity";
        "بيروت";
    }
    "Bishkek";
    {
        "exemplarCity";
        "بشكيك";
    }
    "Brunei";
    {
        "exemplarCity";
        "بروناي";
    }
```

```

    }
    "Calcutta";
    {
        "exemplarCity";
        "كالكتا";
    }
    "Chita";
    {
        "exemplarCity";
        "تشيتا";
    }
    "Choibalsan";
    {
        "exemplarCity";
        "تشوبالسان";
    }
    "Colombo";
    {
        "exemplarCity";
        "كولومبو";
    }
    "Damascus";
    {
        "exemplarCity";
        "دمشق";
    }
    "Dhaka";
    {
        "exemplarCity";
        "دكا";
    }
    "Dili";
    {
        "exemplarCity";
        "ديلي";
    }
    "Dubai";
    {
        "exemplarCity";
        "دبي";
    }
    "Dushanbe";
    {
        "exemplarCity";
        "دوشانبي";
    }
    "Gaza";
    {
        "exemplarCity";
        "غزة";
    }
    "Hebron";
    {
        "exemplarCity";
        "هيبرون (مدينة الخليل)";
    }
    "Hong_Kong";

```

```
{
    "exemplarCity";
    "هونغ كونغ";
}
"Hovd";
{
    "exemplarCity";
    "هوفد";
}
"Irkutsk";
{
    "exemplarCity";
    "ايركيتسك";
}
"Jakarta";
{
    "exemplarCity";
    "جاكرتا";
}
"Jayapura";
{
    "exemplarCity";
    "جاياپورا";
}
"Jerusalem";
{
    "exemplarCity";
    "القدس";
}
"Kabul";
{
    "exemplarCity";
    "كابول";
}
"Kamchatka";
{
    "exemplarCity";
    "كامشاتكا";
}
"Karachi";
{
    "exemplarCity";
    "كراتشي";
}
"Katmandu";
{
    "exemplarCity";
    "كاتماندو";
}
"Khandyga";
{
    "exemplarCity";
    "خانديجا";
}
"Krasnoyarsk";
{
    "exemplarCity";
```

```

        "کراسنویارسک";
    }
    "Kuala_Lumpur";
    {
        "exemplarCity";
        "کوالا لامبور";
    }
    "Kuching";
    {
        "exemplarCity";
        "کیشینج";
    }
    "Kuwait";
    {
        "exemplarCity";
        "الکویت";
    }
    "Macau";
    {
        "exemplarCity";
        "ماکاو";
    }
    "Magadan";
    {
        "exemplarCity";
        "مجادن";
    }
    "Makassar";
    {
        "exemplarCity";
        "ماکسار";
    }
    "Manila";
    {
        "exemplarCity";
        "مانیلا";
    }
    "Muscat";
    {
        "exemplarCity";
        "مسقط";
    }
    "Nicosia";
    {
        "exemplarCity";
        "نیکوسیا";
    }
    "Novokuznetsk";
    {
        "exemplarCity";
        "نوفوکوزنتسک";
    }
    "Novosibirsk";
    {
        "exemplarCity";
        "نوفوسبیرسک";
    }
}

```

```

    "Omsk";
    {
        "exemplarCity";
        "أومسك";
    }
    "Oral";
    {
        "exemplarCity";
        "أورال";
    }
    "Phnom_Penh";
    {
        "exemplarCity";
        "بنوم بنه";
    }
    "Pontianak";
    {
        "exemplarCity";
        "بونتيانك";
    }
    "Pyongyang";
    {
        "exemplarCity";
        "بيونغ يانغ";
    }
    "Qatar";
    {
        "exemplarCity";
        "قطر";
    }
    "Qyzylorda";
    {
        "exemplarCity";
        "كيزيلوردا";
    }
    "Rangoon";
    {
        "exemplarCity";
        "رانغون";
    }
    "Riyadh";
    {
        "exemplarCity";
        "الرياض";
    }
    "Saigon";
    {
        "exemplarCity";
        "مدينة هو تشي منه";
    }
    "Sakhalin";
    {
        "exemplarCity";
        "سكالين";
    }
    "Samarkand";
    {

```



```
        "exemplarCity";
        "سمرقند";
    }
    "Seoul";
    {
        "exemplarCity";
        "서울";
    }
    "Shanghai";
    {
        "exemplarCity";
        "شنغهاي";
    }
    "Singapore";
    {
        "exemplarCity";
        "سنغافورة";
    }
    "Srednekolymsk";
    {
        "exemplarCity";
        "سريدنكوليمسك";
    }
    "Taipei";
    {
        "exemplarCity";
        "تايبه";
    }
    "Tashkent";
    {
        "exemplarCity";
        "تاشقند";
    }
    "Tbilisi";
    {
        "exemplarCity";
        "تبليسي";
    }
    "Tehran";
    {
        "exemplarCity";
        "تهران";
    }
    "Thimphu";
    {
        "exemplarCity";
        "تيمفو";
    }
    "Tokyo";
    {
        "exemplarCity";
        "طوكيو";
    }
    "Tomsk";
    {
        "exemplarCity";
        "تومسك";
    }
```

```
}
"Ulaanbaatar";
{
  "exemplarCity";
  "Улаанбаатар";
}
"Urumqi";
{
  "exemplarCity";
  "أرومقي";
}
"Ust-Nera";
{
  "exemplarCity";
  "أوست نيرا";
}
"Vientiane";
{
  "exemplarCity";
  "فيانتيان";
}
"Vladivostok";
{
  "exemplarCity";
  "فلاديفوستك";
}
"Yakutsk";
{
  "exemplarCity";
  "ياكتسك";
}
"Yekaterinburg";
{
  "exemplarCity";
  "يكاترنبيرج";
}
"Yerevan";
{
  "exemplarCity";
  "يريفان";
}
}
"Indian";
{
  "Antananarivo";
  {
    "exemplarCity";
    "أنتاناناريفو";
  }
  "Chagos";
  {
    "exemplarCity";
    "تشاغوس";
  }
  "Christmas";
  {
    "exemplarCity";
```

```

        "كريسماس";
    }
    "Cocos";
    {
        "exemplarCity";
        "كوكوس";
    }
    "Comoro";
    {
        "exemplarCity";
        "جزر القمر";
    }
    "Kerguelen";
    {
        "exemplarCity";
        "كيرغويلين";
    }
    "Mahe";
    {
        "exemplarCity";
        "ماهي";
    }
    "Maldives";
    {
        "exemplarCity";
        "المالديف";
    }
    "Mauritius";
    {
        "exemplarCity";
        "موريشيوس";
    }
    "Mayotte";
    {
        "exemplarCity";
        "مايوت";
    }
    "Reunion";
    {
        "exemplarCity";
        "ريونيون";
    }
}
"Australia";
{
    "Adelaide";
    {
        "exemplarCity";
        "أديليد";
    }
    "Brisbane";
    {
        "exemplarCity";
        "برسبان";
    }
    "Broken_Hill";
    {

```

```

        "exemplarCity";
        "بروكن هيل";
    }
    "Currie";
    {
        "exemplarCity";
        "كوري";
    }
    "Darwin";
    {
        "exemplarCity";
        "دارون";
    }
    "Eucla";
    {
        "exemplarCity";
        "أوكلا";
    }
    "Hobart";
    {
        "exemplarCity";
        "هوبارت";
    }
    "Lindeman";
    {
        "exemplarCity";
        "ليندمان";
    }
    "Lord_Howe";
    {
        "exemplarCity";
        "لورد هاو";
    }
    "Melbourne";
    {
        "exemplarCity";
        "ميلبورن";
    }
    "Perth";
    {
        "exemplarCity";
        "برثا";
    }
    "Sydney";
    {
        "exemplarCity";
        "سيدني";
    }
}
"Pacific";
{
    "Apia";
    {
        "exemplarCity";
        "أبيا";
    }
    "Auckland";

```

```
{
  "exemplarCity";
  "أوكلاند";
}
"Bougainville";
{
  "exemplarCity";
  "بوغانفيل";
}
"Chatham";
{
  "exemplarCity";
  "تشاثام";
}
"Easter";
{
  "exemplarCity";
  "استر";
}
"Efate";
{
  "exemplarCity";
  "إيفات";
}
"Enderbury";
{
  "exemplarCity";
  "اندربيرج";
}
"Fakaofo";
{
  "exemplarCity";
  "فاكاوفو";
}
"Fiji";
{
  "exemplarCity";
  "فيجي";
}
"Funafuti";
{
  "exemplarCity";
  "فونافوتي";
}
"Galapagos";
{
  "exemplarCity";
  "جلاباجوس";
}
"Gambier";
{
  "exemplarCity";
  "جامبير";
}
"Guadalcanal";
{
  "exemplarCity";
```

```

        "غواد الكانال";
    }
    "Guam";
    {
        "exemplarCity";
        "غوام";
    }
    "Honolulu";
    {
        "exemplarCity";
        "هونولولو";
    }
    "Johnston";
    {
        "exemplarCity";
        "جونستون";
    }
    "Kiritimati";
    {
        "exemplarCity";
        "كيريتي ماتي";
    }
    "Kosrae";
    {
        "exemplarCity";
        "كوسرا";
    }
    "Kwajalein";
    {
        "exemplarCity";
        "كواجالين";
    }
    "Majuro";
    {
        "exemplarCity";
        "ماجورو";
    }
    "Marquesas";
    {
        "exemplarCity";
        "ماركيساس";
    }
    "Midway";
    {
        "exemplarCity";
        "ميدواي";
    }
    "Nauru";
    {
        "exemplarCity";
        "ناورو";
    }
    "Niue";
    {
        "exemplarCity";
        "نيوي";
    }
}

```

```
"Norfolk";
{
  "exemplarCity";
  "نورفولك";
}
"Noumea";
{
  "exemplarCity";
  "نوميا";
}
"Pago_Pago";
{
  "exemplarCity";
  "باغو باغو";
}
"Palau";
{
  "exemplarCity";
  "بالاو";
}
"Pitcairn";
{
  "exemplarCity";
  "بيتكيرن";
}
"Ponape";
{
  "exemplarCity";
  "باناب";
}
"Port_Moresby";
{
  "exemplarCity";
  "بور مورسبي";
}
"Rarotonga";
{
  "exemplarCity";
  "راروتونغا";
}
"Saipan";
{
  "exemplarCity";
  "سايبان";
}
"Tahiti";
{
  "exemplarCity";
  "تاهيتي";
}
"Tarawa";
{
  "exemplarCity";
  "تاراوا";
}
"Tongatapu";
{
```

```

        "exemplarCity";
        "تونغاتابو";
    }
    "Truk";
    {
        "exemplarCity";
        "ترك";
    }
    "Wake";
    {
        "exemplarCity";
        "واك";
    }
    "Wallis";
    {
        "exemplarCity";
        "واليس";
    }
}
"Arctic";
{
    "Longyearbyen";
    {
        "exemplarCity";
        "لونجيربين";
    }
}
"Antarctica";
{
    "Casey";
    {
        "exemplarCity";
        "كاساي";
    }
    "Davis";
    {
        "exemplarCity";
        "دافيز";
    }
    "DumontDUrville";
    {
        "exemplarCity";
        "دي مونت دو روفيل";
    }
    "Macquarie";
    {
        "exemplarCity";
        "ماكوارى";
    }
    "Mawson";
    {
        "exemplarCity";
        "ماوسون";
    }
    "McMurdo";
    {
        "exemplarCity";
    }
}

```



```

        "ماك موردو";
    }
    "Palmer";
    {
        "exemplarCity";
        "بالمير";
    }
    "Rothera";
    {
        "exemplarCity";
        "روثيرا";
    }
    "Syowa";
    {
        "exemplarCity";
        "سايووا";
    }
    "Troll";
    {
        "exemplarCity";
        "ترول";
    }
    "Vostok";
    {
        "exemplarCity";
        "فوستوك";
    }
}
"Etc";
{
    "GMT";
    {
        "exemplarCity";
        "GMT";
    }
    "GMT1";
    {
        "exemplarCity";
        "GMT+1";
    }
    "GMT10";
    {
        "exemplarCity";
        "GMT+10";
    }
    "GMT11";
    {
        "exemplarCity";
        "GMT+11";
    }
    "GMT12";
    {
        "exemplarCity";
        "GMT+12";
    }
    "GMT2";
    {

```

```
        "exemplarCity";
        "GMT+2";
    }
    "GMT3";
    {
        "exemplarCity";
        "GMT+3";
    }
    "GMT4";
    {
        "exemplarCity";
        "GMT+4";
    }
    "GMT5";
    {
        "exemplarCity";
        "GMT+5";
    }
    "GMT6";
    {
        "exemplarCity";
        "GMT+6";
    }
    "GMT7";
    {
        "exemplarCity";
        "GMT+7";
    }
    "GMT8";
    {
        "exemplarCity";
        "GMT+8";
    }
    "GMT9";
    {
        "exemplarCity";
        "GMT+9";
    }
    "GMT-1";
    {
        "exemplarCity";
        "GMT-1";
    }
    "GMT-10";
    {
        "exemplarCity";
        "GMT-10";
    }
    "GMT-11";
    {
        "exemplarCity";
        "GMT-11";
    }
    "GMT-12";
    {
        "exemplarCity";
        "GMT-12";
    }
```

```

    }
    "GMT-13";
    {
        "exemplarCity";
        "GMT-13";
    }
    "GMT-14";
    {
        "exemplarCity";
        "GMT-14";
    }
    "GMT-2";
    {
        "exemplarCity";
        "GMT-2";
    }
    "GMT-3";
    {
        "exemplarCity";
        "GMT-3";
    }
    "GMT-4";
    {
        "exemplarCity";
        "GMT-4";
    }
    "GMT-5";
    {
        "exemplarCity";
        "GMT-5";
    }
    "GMT-6";
    {
        "exemplarCity";
        "GMT-6";
    }
    "GMT-7";
    {
        "exemplarCity";
        "GMT-7";
    }
    "GMT-8";
    {
        "exemplarCity";
        "GMT-8";
    }
    "GMT-9";
    {
        "exemplarCity";
        "GMT-9";
    }
    "Unknown";
    {
        "exemplarCity";
        "مدينة غير معروفة";
    }
}

```

```

    }
    "metazone";
    {
        "Afghanistan";
        {
            "long";
            {
                "standard";
                "توقيت أفغانستان";
            }
        }
        "Africa_Central";
        {
            "long";
            {
                "standard";
                "توقيت وسط أفريقيا";
            }
        }
        "Africa_Eastern";
        {
            "long";
            {
                "standard";
                "توقيت شرق أفريقيا";
            }
        }
        "Africa_Southern";
        {
            "long";
            {
                "standard";
                "توقيت جنوب أفريقيا";
            }
        }
        "Africa_Western";
        {
            "long";
            {
                "generic";
                "توقيت غرب أفريقيا",
                "standard";
                "توقيت غرب أفريقيا الرسمي",
                "daylight";
                "توقيت غرب أفريقيا الصيفي";
            }
        }
        "Alaska";
        {
            "long";
            {
                "generic";
                "توقيت ألاسكا",
                "standard";
                "التوقيت الرسمي لألاسكا",
                "daylight";
                "توقيت ألاسكا الصيفي";
            }
        }
    }

```

```

    }
  }
  "Amazon";
  {
    "long";
    {
      "generic";
      "توقيت الأمازون",
      "standard";
      "توقيت الأمازون الرسمي",
      "daylight";
      "توقيت الأمازون الصيفي";
    }
  }
  "America_Central";
  {
    "long";
    {
      "generic";
      "التوقيت المركزي لأمريكا الشمالية",
      "standard";
      "التوقيت الرسمي المركزي لأمريكا الشمالية",
      "daylight";
      "التوقيت الصيفي المركزي لأمريكا الشمالية";
    }
  }
  "America_Eastern";
  {
    "long";
    {
      "generic";
      "التوقيت الشرقي لأمريكا الشمالية",
      "standard";
      "التوقيت الرسمي الشرقي لأمريكا الشمالية",
      "daylight";
      "التوقيت الصيفي الشرقي لأمريكا الشمالية";
    }
  }
  "America_Mountain";
  {
    "long";
    {
      "generic";
      "التوقيت الجبلي لأمريكا الشمالية",
      "standard";
      "التوقيت الجبلي الرسمي لأمريكا الشمالية",
      "daylight";
      "التوقيت الجبلي الصيفي لأمريكا الشمالية";
    }
  }
  "America_Pacific";
  {
    "long";
    {
      "generic";
      "توقيت المحيط الهادي",
      "standard";

```

```

        "توقيت المحيط الهادي الرسمي",
        "daylight";
        "توقيت المحيط الهادي الصيفي";
    }
}
"Anadyr";
{
    "long";
    {
        "generic";
        "توقيت أنادير",
        "standard";
        "توقيت أنادير الرسمي",
        "daylight";
        "التوقيت الصيفي لأنادير";
    }
}
"Apia";
{
    "long";
    {
        "generic";
        "توقيت آبيا",
        "standard";
        "التوقيت الرسمي لآبيا",
        "daylight";
        "التوقيت الصيفي لآبيا";
    }
}
"Arabian";
{
    "long";
    {
        "generic";
        "التوقيت العربي",
        "standard";
        "التوقيت العربي الرسمي",
        "daylight";
        "التوقيت العربي الصيفي";
    }
}
"Argentina";
{
    "long";
    {
        "generic";
        "توقيت الأرجنتين",
        "standard";
        "توقيت الأرجنتين الرسمي",
        "daylight";
        "توقيت الأرجنتين الصيفي";
    }
}
"Argentina_Western";
{
    "long";
    {

```

```

        "generic";
        "توقيت غرب الأرجنتين",
        "standard";
        "توقيت غرب الأرجنتين الرسمي",
        "daylight";
        "توقيت غرب الأرجنتين الصيفي";
    }
}
"Armenia";
{
    "long";
    {
        "generic";
        "توقيت أرمينيا",
        "standard";
        "توقيت أرمينيا الرسمي",
        "daylight";
        "توقيت أرمينيا الصيفي";
    }
}
"Atlantic";
{
    "long";
    {
        "generic";
        "توقيت الأطلسي",
        "standard";
        "التوقيت الرسمي الأطلسي",
        "daylight";
        "التوقيت الصيفي الأطلسي";
    }
}
"Australia_Central";
{
    "long";
    {
        "generic";
        "توقيت وسط أستراليا",
        "standard";
        "توقيت وسط أستراليا الرسمي",
        "daylight";
        "توقيت وسط أستراليا الصيفي";
    }
}
"Australia_CentralWestern";
{
    "long";
    {
        "generic";
        "توقيت غرب وسط أستراليا",
        "standard";
        "توقيت غرب وسط أستراليا الرسمي",
        "daylight";
        "توقيت غرب وسط أستراليا الصيفي";
    }
}
"Australia_Eastern";

```

```

    {
      "long";
      {
        "generic";
        "توقيت شرق أستراليا",
        "standard";
        "توقيت شرق أستراليا الرسمي",
        "daylight";
        "توقيت شرق أستراليا الصيفي";
      }
    }
    "Australia_Western";
    {
      "long";
      {
        "generic";
        "توقيت غرب أستراليا",
        "standard";
        "توقيت غرب أستراليا الرسمي",
        "daylight";
        "توقيت غرب أستراليا الصيفي";
      }
    }
    "Azerbaijan";
    {
      "long";
      {
        "generic";
        "توقيت أذربيجان",
        "standard";
        "توقيت أذربيجان الرسمي",
        "daylight";
        "توقيت أذربيجان الصيفي";
      }
    }
    "Azores";
    {
      "long";
      {
        "generic";
        "توقيت أزورس",
        "standard";
        "توقيت أزورس الرسمي",
        "daylight";
        "توقيت أزورس الصيفي";
      }
    }
    "Bangladesh";
    {
      "long";
      {
        "generic";
        "توقيت بنجلاديش",
        "standard";
        "توقيت بنجلاديش الرسمي",
        "daylight";
        "توقيت بنجلاديش الصيفي";
      }
    }

```



```

    }
  }
  "Bhutan";
  {
    "long";
    {
      "standard";
      "توقيت بوتان";
    }
  }
  "Bolivia";
  {
    "long";
    {
      "standard";
      "توقيت بوليفيا";
    }
  }
  "Brasilia";
  {
    "long";
    {
      "generic";
      "توقيت برازيليا",
      "standard";
      "توقيت برازيليا الرسمي",
      "daylight";
      "توقيت برازيليا الصيفي";
    }
  }
  "Brunei";
  {
    "long";
    {
      "standard";
      "توقيت بروناي";
    }
  }
  "Cape_Verde";
  {
    "long";
    {
      "generic";
      "توقيت الرأس الأخضر",
      "standard";
      "توقيت الرأس الأخضر الرسمي",
      "daylight";
      "توقيت الرأس الأخضر الصيفي";
    }
  }
  "Chamorro";
  {
    "long";
    {
      "standard";
      "توقيت تشامورو";
    }
  }

```

```

    }
    "Chatham";
    {
        "long";
        {
            "generic";
            "توقيت تشاتام",
            "standard";
            "توقيت تشاتام الرسمي",
            "daylight";
            "توقيت تشاتام الصيفي";
        }
    }
    "Chile";
    {
        "long";
        {
            "generic";
            "توقيت شيلي",
            "standard";
            "توقيت شيلي الرسمي",
            "daylight";
            "توقيت شيلي الصيفي";
        }
    }
    "China";
    {
        "long";
        {
            "generic";
            "توقيت الصين",
            "standard";
            "توقيت الصين الرسمي",
            "daylight";
            "توقيت الصين الصيفي";
        }
    }
    "Choibalsan";
    {
        "long";
        {
            "generic";
            "توقيت شويبالسان",
            "standard";
            "توقيت شويبالسان الرسمي",
            "daylight";
            "التوقيت الصيفي لشويبالسان";
        }
    }
    "Christmas";
    {
        "long";
        {
            "standard";
            "توقيت جزر الكريسماس";
        }
    }
}

```

```

"Cocos";
{
  "long";
  {
    "standard";
    "توقيت جزر كوكوس";
  }
}
"Colombia";
{
  "long";
  {
    "generic";
    "توقيت كولومبيا",
    "standard";
    "توقيت كولومبيا الرسمي",
    "daylight";
    "توقيت كولومبيا الصيفي";
  }
}
"Cook";
{
  "long";
  {
    "generic";
    "توقيت جزر كوك",
    "standard";
    "توقيت جزر كوك الرسمي",
    "daylight";
    "توقيت جزر كوك الصيفي";
  }
}
"Cuba";
{
  "long";
  {
    "generic";
    "توقيت كوبا",
    "standard";
    "توقيت كوبا الرسمي",
    "daylight";
    "توقيت كوبا الصيفي";
  }
}
"Davis";
{
  "long";
  {
    "standard";
    "توقيت دافيز";
  }
}
"DumontDUrville";
{
  "long";
  {
    "standard";
  }
}

```

```

        "توقيت دي مونت دو روفيل";
    }
}
"East_Timor";
{
    "long";
    {
        "standard";
        "توقيت تيمور الشرقية";
    }
}
"Easter";
{
    "long";
    {
        "generic";
        "توقيت جزيرة استر",
        "standard";
        "توقيت جزيرة استر الرسمي",
        "daylight";
        "توقيت جزيرة استر الصيفي";
    }
}
"Ecuador";
{
    "long";
    {
        "standard";
        "توقيت الإكوادور";
    }
}
"Europe_Central";
{
    "long";
    {
        "generic";
        "توقيت وسط أوروبا",
        "standard";
        "توقيت وسط أوروبا الرسمي",
        "daylight";
        "توقيت وسط أوروبا الصيفي";
    }
}
"Europe_Eastern";
{
    "long";
    {
        "generic";
        "توقيت شرق أوروبا",
        "standard";
        "توقيت شرق أوروبا الرسمي",
        "daylight";
        "توقيت شرق أوروبا الصيفي";
    }
}
"Europe_Further_Eastern";
{

```

```

        "long";
        {
            "standard";
            "التوقيت الأوروبي ( أكثر شرقاً )";
        }
    }
    "Europe_Western";
    {
        "long";
        {
            "generic";
            "توقيت غرب أوروبا",
            "standard";
            "توقيت غرب أوروبا الرسمي",
            "daylight";
            "توقيت غرب أوروبا الصيفي";
        }
    }
    "Falkland";
    {
        "long";
        {
            "generic";
            "توقيت جزر فوكلاند",
            "standard";
            "توقيت جزر فوكلاند الرسمي",
            "daylight";
            "توقيت جزر فوكلاند الصيفي";
        }
    }
    "Fiji";
    {
        "long";
        {
            "generic";
            "توقيت فيجي",
            "standard";
            "توقيت فيجي الرسمي",
            "daylight";
            "توقيت فيجي الصيفي";
        }
    }
    "French_Guiana";
    {
        "long";
        {
            "standard";
            "توقيت غايانا الفرنسية";
        }
    }
    "French_Southern";
    {
        "long";
        {
            "standard";
            "توقيت المقاطعات الفرنسية الجنوبية";
        }
    }
    "والأنتارتيكية";

```

```

    }
  }
  "Galapagos";
  {
    "long";
    {
      "standard";
      "توقيت غلابا غوس";
    }
  }
  "Gambier";
  {
    "long";
    {
      "standard";
      "توقيت جامبير";
    }
  }
  "Georgia";
  {
    "long";
    {
      "generic";
      "توقيت جورجيا",
      "standard";
      "توقيت جورجيا الرسمي",
      "daylight";
      "توقيت جورجيا الصيفي";
    }
  }
  "Gilbert_Islands";
  {
    "long";
    {
      "standard";
      "توقيت جزر جيلبرت";
    }
  }
  "GMT";
  {
    "long";
    {
      "standard";
      "توقيت غرينتش";
    }
  }
  "Greenland_Eastern";
  {
    "long";
    {
      "generic";
      "توقيت شرق غرينلاند",
      "standard";
      "توقيت شرق غرينلاند الرسمي",
      "daylight";
      "توقيت شرق غرينلاند الصيفي";
    }
  }

```

```

    }
    "Greenland_Western";
    {
        "long";
        {
            "generic";
            "توقيت غرب غرينلاند",
            "standard";
            "توقيت غرب غرينلاند الرسمي",
            "daylight";
            "توقيت غرب غرينلاند الصيفي";
        }
    }
    "Guam";
    {
        "long";
        {
            "standard";
            "توقيت غوام";
        }
    }
    "Gulf";
    {
        "long";
        {
            "standard";
            "توقيت الخليج";
        }
    }
    "Guyana";
    {
        "long";
        {
            "standard";
            "توقيت غيانا";
        }
    }
    "Hawaii_Aleutian";
    {
        "long";
        {
            "generic";
            "توقيت هاواي ألوتيان",
            "standard";
            "توقيت هاواي ألوتيان الرسمي",
            "daylight";
            "توقيت هاواي ألوتيان الصيفي";
        }
    }
    "Hong_Kong";
    {
        "long";
        {
            "generic";
            "توقيت هونغ كونغ",
            "standard";
            "توقيت هونغ كونغ الرسمي";
        }
    }

```

```

        "daylight";
        "توقيت هونغ كونغ الصيفي";
    }
}
"Hovd";
{
    "long";
    {
        "generic";
        "توقيت هوفد",
        "standard";
        "توقيت هوفد الرسمي",
        "daylight";
        "توقيت هوفد الصيفي";
    }
}
"India";
{
    "long";
    {
        "standard";
        "توقيت الهند";
    }
}
"Indian_Ocean";
{
    "long";
    {
        "standard";
        "توقيت المحيط الهندي";
    }
}
"Indochina";
{
    "long";
    {
        "standard";
        "توقيت الهند الصينية";
    }
}
"Indonesia_Central";
{
    "long";
    {
        "standard";
        "توقيت وسط إندونيسيا";
    }
}
"Indonesia_Eastern";
{
    "long";
    {
        "standard";
        "توقيت شرق إندونيسيا";
    }
}
"Indonesia_Western";

```



```

    {
      "long";
      {
        "standard";
        "توقيت غرب إندونيسيا";
      }
    }
    "Iran";
    {
      "long";
      {
        "generic";
        "توقيت إيران",
        "standard";
        "توقيت إيران الرسمي",
        "daylight";
        "توقيت إيران الصيفي";
      }
    }
    "Irkutsk";
    {
      "long";
      {
        "generic";
        "توقيت إركوتسك",
        "standard";
        "توقيت إركوتسك الرسمي",
        "daylight";
        "توقيت إركوتسك الصيفي";
      }
    }
    "Israel";
    {
      "long";
      {
        "generic";
        "توقيت إسرائيل",
        "standard";
        "توقيت إسرائيل الرسمي",
        "daylight";
        "توقيت إسرائيل الصيفي";
      }
    }
    "Japan";
    {
      "long";
      {
        "generic";
        "توقيت اليابان",
        "standard";
        "توقيت اليابان الرسمي",
        "daylight";
        "توقيت اليابان الصيفي";
      }
    }
    "Kamchatka";
    {

```

```

        "long";
        {
            "generic";
            "توقيت كامشاتكا",
            "standard";
            "توقيت بيتروبافلوفسك-كامشاتسكي",
            "daylight";
            "توقيت بيتروبافلوفسك-كامشاتسكي الصيفي";
        }
    }
    "Kazakhstan_Eastern";
    {
        "long";
        {
            "standard";
            "توقيت شرق كازاخستان";
        }
    }
    "Kazakhstan_Western";
    {
        "long";
        {
            "standard";
            "توقيت غرب كازاخستان";
        }
    }
    "Korea";
    {
        "long";
        {
            "generic";
            "توقيت كوريا",
            "standard";
            "توقيت كوريا الرسمي",
            "daylight";
            "توقيت كوريا الصيفي";
        }
    }
    "Kosrae";
    {
        "long";
        {
            "standard";
            "توقيت كوسرا";
        }
    }
    "Krasnoyarsk";
    {
        "long";
        {
            "generic";
            "توقيت كراسنويارسك",
            "standard";
            "توقيت كراسنويارسك الرسمي",
            "daylight";
            "التوقيت الصيفي لكراسنويارسك";
        }
    }

```

```

    }
    "Kyrgystan";
    {
        "long";
        {
            "standard";
            "توقيت قرغيزستان";
        }
    }
    "Line_Islands";
    {
        "long";
        {
            "standard";
            "توقيت جزر لاين";
        }
    }
    "Lord_Howe";
    {
        "long";
        {
            "generic";
            "توقيت لورد هاو",
            "standard";
            "توقيت لورد هاو الرسمي",
            "daylight";
            "التوقيت الصيفي للورد هاو";
        }
    }
    "Macquarie";
    {
        "long";
        {
            "standard";
            "توقيت ماكوارى";
        }
    }
    "Magadan";
    {
        "long";
        {
            "generic";
            "توقيت ماغادان",
            "standard";
            "توقيت ماغادان الرسمي",
            "daylight";
            "توقيت ماغادان الصيفي";
        }
    }
    "Malaysia";
    {
        "long";
        {
            "standard";
            "توقيت ماليزيا";
        }
    }
}

```

```

"Maldives";
{
  "long";
  {
    "standard";
    "توقيت المالديف";
  }
}
"Marquesas";
{
  "long";
  {
    "standard";
    "توقيت ماركيساس";
  }
}
"Marshall_Islands";
{
  "long";
  {
    "standard";
    "توقيت جزر مارشال";
  }
}
"Mauritius";
{
  "long";
  {
    "generic";
    "توقيت موريشيوس",
    "standard";
    "توقيت موريشيوس الرسمي",
    "daylight";
    "توقيت موريشيوس الصيفي";
  }
}
"Mawson";
{
  "long";
  {
    "standard";
    "توقيت ماوسون";
  }
}
"Mexico_Northwest";
{
  "long";
  {
    "generic";
    "توقيت شمال غرب المكسيك",
    "standard";
    "التوقيت الرسمي لشمال غرب المكسيك",
    "daylight";
    "التوقيت الصيفي لشمال غرب المكسيك";
  }
}
"Mexico_Pacific";

```

```

{
  "long";
  {
    "generic";
    "توقيت المحيط الهادي للمكسيك",
    "standard";
    "توقيت المحيط الهادي الرسمي للمكسيك",
    "daylight";
    "توقيت المحيط الهادي الصيفي للمكسيك";
  }
}
"Mongolia";
{
  "long";
  {
    "generic";
    "توقيت أولان باتور",
    "standard";
    "توقيت أولان باتور الرسمي",
    "daylight";
    "توقيت أولان باتور الصيفي";
  }
}
"Moscow";
{
  "long";
  {
    "generic";
    "توقيت موسكو",
    "standard";
    "توقيت موسكو الرسمي",
    "daylight";
    "توقيت موسكو الصيفي";
  }
}
"Myanmar";
{
  "long";
  {
    "standard";
    "توقيت ميانمار";
  }
}
"Nauru";
{
  "long";
  {
    "standard";
    "توقيت ناورو";
  }
}
"Nepal";
{
  "long";
  {
    "standard";
    "توقيت نيبال";
  }
}

```

```

    }
  }
  "New_Caledonia";
  {
    "long";
    {
      "generic";
      "توقيت كاليدونيا الجديدة",
      "standard";
      "توقيت كاليدونيا الجديدة الرسمي",
      "daylight";
      "توقيت كاليدونيا الجديدة الصيفي";
    }
  }
  "New_Zealand";
  {
    "long";
    {
      "generic";
      "توقيت نيوزيلندا",
      "standard";
      "توقيت نيوزيلندا الرسمي",
      "daylight";
      "توقيت نيوزيلندا الصيفي";
    }
  }
  "Newfoundland";
  {
    "long";
    {
      "generic";
      "توقيت نيوفاوندلاند",
      "standard";
      "توقيت نيوفاوندلاند الرسمي",
      "daylight";
      "توقيت نيوفاوندلاند الصيفي";
    }
  }
  "Niue";
  {
    "long";
    {
      "standard";
      "توقيت نيوي";
    }
  }
  "Norfolk";
  {
    "long";
    {
      "standard";
      "توقيت جزيرة نورفولك";
    }
  }
  "Noronha";
  {
    "long";

```

```

        {
            "generic";
            "توقيت فيرناندو دي نورونها",
            "standard";
            "توقيت فرناندو دي نورونها الرسمي",
            "daylight";
            "توقيت فرناندو دي نورونها الصيفي";
        }
    }
    "North_Mariana";
    {
        "long";
        {
            "standard";
            "توقيت جزر ماريانا الشمالية";
        }
    }
    "Novosibirsk";
    {
        "long";
        {
            "generic";
            "توقيت نوفوسيبيرسك",
            "standard";
            "توقيت نوفوسيبيرسك الرسمي",
            "daylight";
            "توقيت نوفوسيبيرسك الصيفي";
        }
    }
    "Omsk";
    {
        "long";
        {
            "generic";
            "توقيت أومسك",
            "standard";
            "توقيت أومسك الرسمي",
            "daylight";
            "توقيت أومسك الصيفي";
        }
    }
    "Pakistan";
    {
        "long";
        {
            "generic";
            "توقيت باكستان",
            "standard";
            "توقيت باكستان الرسمي",
            "daylight";
            "توقيت باكستان الصيفي";
        }
    }
    "Palau";
    {
        "long";
        {

```

```

        "standard";
        "توقيت بالاو";
    }
}
"Papua_New_Guinea";
{
    "long";
    {
        "standard";
        "توقيت بابوا غينيا الجديدة";
    }
}
"Paraguay";
{
    "long";
    {
        "generic";
        "توقيت باراغواي",
        "standard";
        "توقيت باراغواي الرسمي",
        "daylight";
        "توقيت باراغواي الصيفي";
    }
}
"Peru";
{
    "long";
    {
        "generic";
        "توقيت بيرو",
        "standard";
        "توقيت بيرو الرسمي",
        "daylight";
        "توقيت بيرو الصيفي";
    }
}
"Philippines";
{
    "long";
    {
        "generic";
        "توقيت الفلبين",
        "standard";
        "توقيت الفلبين الرسمي",
        "daylight";
        "توقيت الفلبين الصيفي";
    }
}
"Phoenix_Islands";
{
    "long";
    {
        "standard";
        "توقيت جزر فينكس";
    }
}
"Pierre_Miquelon";

```



```

    {
      "long";
      {
        "generic";
        "توقيت سانت بيير وميكلون",
        "standard";
        "توقيت سانت بيير وميكلون الرسمي",
        "daylight";
        "توقيت سانت بيير وميكلون الصيفي";
      }
    }
    "Pitcairn";
    {
      "long";
      {
        "standard";
        "توقيت بيتكيرن";
      }
    }
    "Ponape";
    {
      "long";
      {
        "standard";
        "توقيت بونابي";
      }
    }
    "Pyongyang";
    {
      "long";
      {
        "standard";
        "توقيت بيونغ يانغ";
      }
    }
    "Reunion";
    {
      "long";
      {
        "standard";
        "توقيت ريونيون";
      }
    }
    "Rothera";
    {
      "long";
      {
        "standard";
        "توقيت روثيرا";
      }
    }
    "Sakhalin";
    {
      "long";
      {
        "generic";
        "توقيت ساخالين",

```

```

        "standard";
        "توقيت ساخالين الرسمي",
        "daylight";
        "توقيت ساخالين الصيفي";
    }
}
"Samara";
{
    "long";
    {
        "generic";
        "توقيت سامارا",
        "standard";
        "توقيت سامارا",
        "daylight";
        "توقيت سامارا الصيفي";
    }
}
"Samoa";
{
    "long";
    {
        "generic";
        "توقيت ساموا",
        "standard";
        "توقيت ساموا الرسمي",
        "daylight";
        "توقيت ساموا الصيفي";
    }
}
"Seychelles";
{
    "long";
    {
        "standard";
        "توقيت سيشل";
    }
}
"Singapore";
{
    "long";
    {
        "standard";
        "توقيت سنغافورة";
    }
}
"Solomon";
{
    "long";
    {
        "standard";
        "توقيت جزر سليمان";
    }
}
"South_Georgia";
{
    "long";

```

```

        {
            "standard";
            "توقيت جنوب جورجيا";
        }
    }
    "Suriname";
    {
        "long";
        {
            "standard";
            "توقيت سورينام";
        }
    }
    "Syowa";
    {
        "long";
        {
            "standard";
            "توقيت سايووا";
        }
    }
    "Tahiti";
    {
        "long";
        {
            "standard";
            "توقيت تاهيتي";
        }
    }
    "Taipei";
    {
        "long";
        {
            "generic";
            "توقيت تايبه",
            "standard";
            "توقيت تايبه الرسمي",
            "daylight";
            "توقيت تايبه الصيفي";
        }
    }
    "Tajikistan";
    {
        "long";
        {
            "standard";
            "توقيت طاجكستان";
        }
    }
    "Tokelau";
    {
        "long";
        {
            "standard";
            "توقيت توكيلاو";
        }
    }
}

```

```

"Tonga";
{
  "long";
  {
    "generic";
    "توقيت تونغا",
    "standard";
    "توقيت تونغا الرسمي",
    "daylight";
    "توقيت تونغا الصيفي";
  }
}
"Truk";
{
  "long";
  {
    "standard";
    "توقيت شوك";
  }
}
"Turkmenistan";
{
  "long";
  {
    "generic";
    "توقيت تركمانستان",
    "standard";
    "توقيت تركمانستان الرسمي",
    "daylight";
    "توقيت تركمانستان الصيفي";
  }
}
"Tuvalu";
{
  "long";
  {
    "standard";
    "توقيت توفالو";
  }
}
"Uruguay";
{
  "long";
  {
    "generic";
    "توقيت أورغواي",
    "standard";
    "توقيت أورغواي الرسمي",
    "daylight";
    "توقيت أورغواي الصيفي";
  }
}
"Uzbekistan";
{
  "long";
  {
    "generic";

```

```

        "توقيت أوزبكستان",
        "standard";
        "توقيت أوزبكستان الرسمي",
        "daylight";
        "توقيت أوزبكستان الصيفي";
    }
}
"Vanuatu";
{
    "long";
    {
        "generic";
        "توقيت فانواتو",
        "standard";
        "توقيت فانواتو الرسمي",
        "daylight";
        "توقيت فانواتو الصيفي";
    }
}
"Venezuela";
{
    "long";
    {
        "standard";
        "توقيت فنزويلا";
    }
}
"Vladivostok";
{
    "long";
    {
        "generic";
        "توقيت فلاديفوستوك",
        "standard";
        "توقيت فلاديفوستوك الرسمي",
        "daylight";
        "توقيت فلاديفوستوك الصيفي";
    }
}
"Volgograd";
{
    "long";
    {
        "generic";
        "توقيت فولغوغراد",
        "standard";
        "توقيت فولغوغراد الرسمي",
        "daylight";
        "توقيت فولغوغراد الصيفي";
    }
}
"Vostok";
{
    "long";
    {
        "standard";
        "توقيت فوستوك";
    }
}

```

```

    }
    "Wake";
    {
        "long";
        {
            "standard";
            "توقيت جزيرة ويك";
        }
    }
    "Wallis";
    {
        "long";
        {
            "standard";
            "توقيت واليس و فوتونا";
        }
    }
    "Yakutsk";
    {
        "long";
        {
            "generic";
            "توقيت ياكوتسك",
            "standard";
            "توقيت ياكوتسك الرسمي",
            "daylight";
            "توقيت ياكوتسك الصيفي";
        }
    }
    "Yekaterinburg";
    {
        "long";
        {
            "generic";
            "توقيت يكاترينبورغ",
            "standard";
            "توقيت يكاترينبورغ الرسمي",
            "daylight";
            "توقيت يكاترينبورغ الصيفي";
        }
    }
}
}
}
}
}
}
}
}
}
}

```

Customization in React Calendar component

Calendar allows you to customize the entire appearance by using the custom CSS and [renderDayCell](#) event to customize the each day cell.

This following section demonstrates how to disable, highlights the specific dates in the Calendar.

Disable Weekends

You can disable the weekends of every month in a Calendar by using the [renderDayCell](#) event. The `isDisabled` argument from this event allows you to define whether the date to be disabled or not.

Set `isDisabled` to true to disable the date value.

The following example demonstrates how to disable weekends of every month.

[Class-component]

INDEX.JSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    // initialize the value
    dateValue = new Date();
    disabledDate(args) {
        if (args.date.getDay() === 0 || args.date.getDay() === 6) {
            // set 'true' to disable the weekends
            args.isDisabled = true;
        }
    }
    render() {
        return <CalendarComponent id="calendar"
renderDayCell={this.disabledDate} value={this.dateValue}/>;
    }
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the calendarcomponent
import { CalendarComponent, RenderDayCellEventArgs } from '@syncfusion/ej2-
react-calendars';
import * as React from "react";
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // initialize the value
    private dateValue: Date = new Date();
    public disabledDate(args: RenderDayCellEventArgs): void {
        if ((args.date as Date).getDay() === 0 || (args.date as
Date).getDay() === 6) {
            // set 'true' to disable the weekends
            args.isDisabled = true;
        }
    }
    public render() {
        return <CalendarComponent id="calendar"
renderDayCell={this.disabledDate} value={this.dateValue} />
    }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]**INDEX.JSX**

```
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from 'react-dom';
function App() {
    // initialize the value
    let dateValue = new Date();
    function disabledDate(args) {
        if (args.date.getDay() === 0 || args.date.getDay() === 6) {
            // set 'true' to disable the weekends
            args.isDisabled = true;
        }
    }
    return <CalendarComponent id="calendar" renderDayCell={disabledDate}
value={dateValue}/>;
};
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import { CalendarComponent, RenderDayCellEventArgs } from '@syncfusion/ej2-
react-calendars';
import * as React from "react";
import * as ReactDOM from 'react-dom';
function App() {
    // initialize the value
    let dateValue: Date = new Date();
    function disabledDate(args: RenderDayCellEventArgs): void {
        if ((args.date as Date).getDay() === 0 || (args.date as
Date).getDay() === 6) {
            // set 'true' to disable the weekends
            args.isDisabled = true;
        }
    }
    return <CalendarComponent id="calendar" renderDayCell={disabledDate}
value={dateValue} />;
};
ReactDOM.render(<App />, document.getElementById('element'));
```

Day Cell Format

You can highlight the specific dates by adding the custom CSS or element to the day cell by using [renderDayCell](#).

Below is the list of classes that provides the flexible way to customize the Calendar component.

Class Name	Description
---	---
e-calendar	Applied to Calendar.
e-header	Applied to header.

- | e-title | Applied to title. |
- | e-icon-container | Applied to previous and next icon container. |
- | e-prev | Applied to previous icon. |
- | e-next | Applied to next icon. |
- | e-weekend | Applied to weekend dates. |
- | e-other-month | Applied to other month dates. |
- | e-day | Applied to each day cell. |
- | e-selected | Applied to selected dates. |
- | e-disabled | Applied to disabled dates. |

The following example highlights the world health date (7th April every year) and world forest day (21st March every year) in the Calendar by using the custom icon and tooltip.

[Class-component]

INDEX.JSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  // initialize the value
  dateValue = new Date('03/10/2017');
  customDates(args) {
    let span;
    // defines the custom HTML element to be added.
    span = document.createElement('span');
    // Use "e-icons" class name to load Essential JS 2 built-in icons.
    span.setAttribute('class', 'e-icons highlight-day');
    if (args.date.getDate() === 7 && args.date.getMonth() === 3) {
      // append the span element to day cell.
      args.element.appendChild(span);
      // set the custom tooltip to the special dates.
      args.element.setAttribute('title', 'World health day!');
      // Use "special" class name to highlight the special dates,
      // which you can refer in "styles.css".
      args.element.className = 'special';
    }
    if (args.date.getDate() === 21 && args.date.getMonth() === 2) {
      args.element.appendChild(span);
      args.element.className = 'special';
      // set the custom tooltip to the special dates.
      args.element.setAttribute('title', 'World forest day');
    }
  }
  render() {
    return <CalendarComponent id="calendar"
    renderDayCell={this.customDates} value={this.dateValue}/>;
  }
}
```

```
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the calendarcomponent
import { CalendarComponent, RenderDayCellEventArgs } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}>, {}> {
    // initialize the value
    private dateValue: Date = new Date('03/10/2017');
    public customDates(args: RenderDayCellEventArgs): void {
        let span: HTMLElement;
        // defines the custom HTML element to be added.
        span = document.createElement('span');
        // Use "e-icons" class name to load Essential JS 2 built-in icons.
        span.setAttribute('class', 'e-icons highlight-day');
        if ((args.date as Date).getDate() === 7 && (args.date as
Date).getMonth() === 3) {
            // append the span element to day cell.
            (args.element as HTMLElement).appendChild(span);
            // set the custom tooltip to the special dates.
            (args.element as HTMLElement).setAttribute('title', 'World
health day!');
            // Use "special" class name to highlight the special dates,
which you can refer in "styles.css".
            (args.element as HTMLElement).className = 'special';
        }
        if ((args.date as Date).getDate() === 21 && (args.date as
Date).getMonth() === 2) {
            (args.element as HTMLElement).appendChild(span);
            (args.element as HTMLElement).className = 'special';
            // set the custom tooltip to the special dates.
            (args.element as HTMLElement).setAttribute('title', 'World
forest day!');
        }
    }
    public render() {
        return <CalendarComponent id="calendar"
renderDayCell={this.customDates} value={this.dateValue} />
    }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]

INDEX.JSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from 'react-dom';
function App() {
    // initialize the value
    const dateValue = new Date('03/10/2017');
```

```

function customDates(args) {
    let span;
    // defines the custom HTML element to be added.
    span = document.createElement('span');
    // Use "e-icons" class name to load Essential JS 2 built-in icons.
    span.setAttribute('class', 'e-icons highlight-day');
    if (args.date.getDate() === 7 && args.date.getMonth() === 3) {
        // append the span element to day cell.
        args.element.appendChild(span);
        // set the custom tooltip to the special dates.
        args.element.setAttribute('title', 'World health day!');
        // Use "special" class name to highlight the special dates,
        // which you can refer in "styles.css".
        args.element.className = 'special';
    }
    if (args.date.getDate() === 21 && args.date.getMonth() === 2) {
        args.element.appendChild(span);
        args.element.className = 'special';
        // set the custom tooltip to the special dates.
        args.element.setAttribute('title', 'World forest day');
    }
}

return <CalendarComponent id="calendar" renderDayCell={customDates}
value={dateValue}/>;
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

// import the calendarcomponent
import { CalendarComponent, RenderDayCellEventArgs } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from 'react-dom';
function App() {
    // initialize the value
    const dateValue: Date = new Date('03/10/2017');
    function customDates(args: RenderDayCellEventArgs): void {
        let span: HTMLElement;
        // defines the custom HTML element to be added.
        span = document.createElement('span');
        // Use "e-icons" class name to load Essential JS 2 built-in icons.
        span.setAttribute('class', 'e-icons highlight-day');
        if ((args.date as Date).getDate() === 7 && (args.date as
Date).getMonth() === 3) {
            // append the span element to day cell.
            (args.element as HTMLElement).appendChild(span);
            // set the custom tooltip to the special dates.
            (args.element as HTMLElement).setAttribute('title', 'World
health day!');
            // Use "special" class name to highlight the special dates,
            // which you can refer in "styles.css".
            (args.element as HTMLElement).className = 'special';
        }
    }
}

```

```

        if ((args.date as Date).getDate() === 21 && (args.date as
Date).getMonth() === 2) {
            (args.element as HTMLElement).appendChild(span);
            (args.element as HTMLElement).className = 'special';
            // set the custom tooltip to the special dates.
            (args.element as HTMLElement).setAttribute('title', 'World
forest day');
        }
    }
    return <CalendarComponent id="calendar" renderDayCell={customDates}
value={dateValue} />
};
ReactDOM.render(<App />, document.getElementById('element'));

```

Highlight Weekends

You can highlight the weekends of every month in a Calendar by using the [renderDayCell](#) event. The following example demonstrates how to highlights the weekends of every month.

[Class-component]

INDEX.JSX

```

// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
    // initialize the value
    dateValue = new Date();
    highlightWeekend(args) {
        if (args.date.getDay() === 0 || args.date.getDay() === 6) {
            // To highlight the week end of every month
            args.element.classList.add('e-highlightweekend');
        }
    }
    render() {
        return <CalendarComponent id="calendar"
renderDayCell={this.highlightWeekend} value={this.dateValue}/>;
    }
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

// import the calendarcomponent
import { CalendarComponent, RenderDayCellEventArgs } from '@syncfusion/ej2-
react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
    // initialize the value
    private dateValue: Date = new Date();
    public highlightWeekend(args: RenderDayCellEventArgs): void {
        if ((args.date as Date).getDay() === 0 || (args.date as
Date).getDay() === 6) {

```

```

        // To highlight the week end of every month
        args.element.classList.add('e-highlightweekend');
    }
}
public render() {
    return <CalendarComponent id="calendar"
renderDayCell={this.highlightWeekend} value={this.dateValue} />
}
};
ReactDOM.render(<App />, document.getElementById('element'));

```

[Functional-component]

INDEX.JSX

```

// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    // initialize the value
    const dateValue = new Date();
    function highlightWeekend(args) {
        if (args.date.getDay() === 0 || args.date.getDay() === 6) {
            // To highlight the week end of every month
            args.element.classList.add('e-highlightweekend');
        }
    }
    return <CalendarComponent id="calendar" renderDayCell={highlightWeekend}
value={dateValue}/>;
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

// import the calendarcomponent
import { CalendarComponent, RenderDayCellEventArgs } from '@syncfusion/ej2-
react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    // initialize the value
    const dateValue: Date = new Date();
    function highlightWeekend(args: RenderDayCellEventArgs): void {
        if ((args.date as Date).getDay() === 0 || (args.date as
Date).getDay() === 6) {
            // To highlight the week end of every month
            args.element.classList.add('e-highlightweekend');
        }
    }
    return <CalendarComponent id="calendar" renderDayCell={highlightWeekend}
value={dateValue} />
};
ReactDOM.render(<App />, document.getElementById('element'));

```

See Also

- [Add the external button in the Calendar popup](#)
- [How to skip a month in Calendar](#)
- [How to change the first day of week](#)
- [How to customize the Calendar day header](#)

Multi select in React Calendar component

Calendar provides an option to select **single** or **multiple dates** by using `isMultiSelection` and `values` properties. By default, `isMultiSelection` property will be in disabled state.

| API | Type | Description |

|-----|-----|-----|

| `isMultiSelection` | **Boolean** | Enables the multi selection option in the Calendar control |

| `values` | **Date[]** | Gets or sets the date range values in multi selection option |

The following example demonstrates the functionality of `isMultiSelection` property and `values` properties in the Calendar control.

[Class-component]

INDEX.JSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
    //initialize the values
    values = [new Date('1/1/2020'), new Date('1/15/2020'), new
Date('1/3/2020'), new Date('1/25/2020')];
    render() {
        return <CalendarComponent id="calendar" isMultiSelection={true}
values={this.values}/>;
    }
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
    //initialize the values
    private values: Date[] = [new Date('1/1/2020'), new Date('1/15/2020'), new
Date('1/3/2020'), new Date('1/25/2020')];
    public render() {
        return <CalendarComponent id="calendar" isMultiSelection={true}
values={this.values} />
    }
}
```

```

    }
  };
  ReactDOM.render(<App />, document.getElementById('element'));

```

[Functional-component]

INDEX.JSX

```

import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  //initialize the values
  const values = [new Date('1/1/2020'), new Date('1/15/2020'), new
Date('1/3/2020'), new Date('1/25/2020')];
  return <CalendarComponent id="calendar" isMultiSelection={true}
values={values}/>;
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { CalendarComponent} from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  //initialize the values
  const values: Date[] = [new Date('1/1/2020'), new Date('1/15/2020'), new
Date('1/3/2020'), new Date('1/25/2020')];
  return <CalendarComponent id="calendar" isMultiSelection={true}
values={values} />
};
ReactDOM.render(<App />, document.getElementById('element'));

```

Calendar views in React Calendar component

The Calendar has the following predefined views that provides a flexible way to navigate back and forth to select the date. Use the [start](#) property to change the default view of the Calendar.

| View | Description |

| --- | --- |

| month (default) | Displays the days in a month |

| year | Displays the months in a year |

| decade | Displays the years in a decade |

The following example demonstrates how to set the **year** as the start view of the calendar.

[Class-component]

INDEX.JSX

```

// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';

```

```
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
  render() {
    //creates a calendar with decade view and navigate up to year view.
    return <CalendarComponent id="calendar" start='Year'/>;
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
  public render() {
    //creates a calendar with decade view and navigate up to year view.
    return <CalendarComponent id="calendar" start='Year' />
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]**INDEX.JSX**

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  //creates a calendar with decade view and navigate up to year view.
  return <CalendarComponent id="calendar" start="Year"/>;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  //creates a calendar with decade view and navigate up to year view.
  return <CalendarComponent id="calendar" start="Year" />;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

View Restriction

Calendar view can be restricted by defining the [start](#) and [depth](#)

By defining the start and depth property with the different view, drill-down and drill-up views navigation can be limited to the user. Calendar views will be drill-down up to the view which is set in **depth** property and drill-up up to the view which is set in **start** property.

Always the depth view has to be smaller than the start view.

[Class-component]

INDEX.JSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
  render() {
    // creates a calendar with decade view and navigate up to year view.
    return <CalendarComponent id="calendar" start='Decade'
    depth='Year' />;
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
  public render() {
    // creates a calendar with decade view and navigate up to year view.
    return <CalendarComponent id="calendar" start='Decade' depth='Year'
    />
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]

INDEX.JSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  // creates a calendar with decade view and navigate up to year view.
  return <CalendarComponent id="calendar" start="Decade" depth="Year"/>;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the calendarcomponent
```

```
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  // creates a calendar with decade view and navigate up to year view.
  return <CalendarComponent id="calendar" start="Decade" depth="Year" />;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

You can restrict the calendar's drill down navigation by defining the [start](#) and [depth](#) property

The following example demonstrates how to select the date in **year** view.

[Class-component]

INDEX.JSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
  render() {
    //creates a calendar with decade view and navigate up to year view.
    return <CalendarComponent id="calendar" start='Year' depth='Year'/>;
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
  public render() {
    //creates a calendar with decade view and navigate up to year view.
    return <CalendarComponent id="calendar" start='Year' depth='Year' />;
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]

INDEX.JSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  //creates a calendar with decade view and navigate up to year view.
  return <CalendarComponent id="calendar" start="Year" depth="Year"/>;
}
```

```
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  //creates a calendar with decade view and navigate up to year view.
  return <CalendarComponent id="calendar" start="Year" depth="Year" />;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

Accessibility in React Calendar component

The Calendar component followed the accessibility guidelines and standards, including [ADA, Section 508](#), [WCAG 2.2](#) standards, and [WCAG roles](#) that are commonly used to evaluate accessibility.

The accessibility compliance for the Calendar component is outlined below.

| Accessibility Criteria | Compatibility |

| -- | -- |

| [WCAG 2.2 Support](#) | |

| [Section 508 Support](#) | |

| [Screen Reader Support](#) | |

| [Right-To-Left Support](#) | |

| [Color Contrast](#) | |

| [Mobile Device Support](#) | |

| [Keyboard Navigation Support](#) | |

| [Accessibility Checker Validation](#) | |

| [Axe-core Accessibility Validation](#) | |

<style>

.post .post-content img {

display: inline-block;

margin: 0.5em 0;

```

}
</style>

<div> - All
features of the component meet the requirement.</div>

<div> - Some features of the component do not meet the requirement.</div>

<div> - The component does not meet the requirement.</div>

```

WAI-ARIA attributes

The Web accessibility defines a way to make web content and web applications more accessible to disabled people. It especially helps the dynamic content change and advanced user interface controls developed with Ajax, HTML, JavaScript, and related technologies.

Calendar provides built-in compliance with the [WAI-ARIA](#) specifications. WAI-ARIA supports is achieved through the attributes like `aria-label`, `aria-selected`, `aria-disabled`, `aria-activedescendant` applied for navigation buttons, disable and active day cells.

It helps to provides information about the widget for assistive technology to the disabled person in the screen reader. Calendar component contains grid as role and grid cell for each day cell

- **Aria-label** : attribute provides the text label for an object for the previous and next month element. It helps the screen reader object to read for the assistive purpose.
- **Aria-selected** : attribute indicates the currently selected date of the Calendar component.
- **Aria-disabled** : attribute indicates the disabled state of this Calendar component.
- **Aria-activedescendent** : attribute helps in managing the current active child of the Calendar component.
- **Role** : attributes gives assistive technologies information about how to handle each element in a widget.
- **Grid-cell** : attributes define the individual cell that can be focusable and selectable.

Keyboard Interaction

You can use the following keys to interact with the Calendar. The component implements the keyboard navigation support by following the [WAI-ARIA practices](#)

It supports the below list of shortcut keys.

| **Press** | **To do this** |

| --- | --- |

| **Upper Arrow** | **Focus the previous week date.** |

| **Down Arrow** | **Focus the next week date.** |

| **Left Arrow** | **Focus the previous date.** |

| **Right Arrow** | **Focus the next date.** |

| **Home** | **Focus the first date in the month.** |

| **End** | **Focus the last date in the month.** |

- | Page Up | Focus the same date in the previous month. |
- | Page Down | Focus the same date in the next month. |
- | Enter | Select the currently focused date. |
- | Shift + Page Up | Focus the same date in the previous year. |
- | Shift + Page Down | Focus the same date in the next year. |
- | Control + Upper Arrow | Moves into the inner level of view like month-year, year-decade |
- | Control + Down Arrow | Moves out from the depth level view like decade-year, year-month |
- | Control + Home | Focus the starting date in the current year. |
- | Control + End | Focus the ending date in the current year. |

To focus the Calendar component use the `alt+t` keys.

[Class-component]

INDEX.JSX

```
{% raw %}
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
  calendarInstance;
  componentDidMount() {
    const proxy = this;
    document.onkeyup = (e) => {
      if (e.altKey && e.keyCode === 84 /* t */) {
        // press altt to focus the control.
      }
    };
    proxy.calendarInstance.element.querySelector('table').focus();
  }
  render() {
    return (
      // specifies the tag for render the Calendar component
      <CalendarComponent id="calendar" ref={ (scope) => {
        this.calendarInstance = scope; } } />);
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import { Calendar, CalendarComponent } from '@syncfusion/ej2-react-
calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
```

```

private calendarInstance: Calendar;
public componentDidMount() {
    const proxy = this;
    document.onkeyup = (e: KeyboardEvent) => {
        if (e.altKey && e.keyCode === 84 /* t */) {
            // press altt to focus the control.
            (proxy.calendarInstance.element.querySelector('table') as
HTMLInputElement).focus();
        }
    };
}
public render() {
    return (
        // specifies the tag for render the Calendar component
        <CalendarComponent id="calendar" ref={(scope) => {
        (this.calendarInstance as Calendar | null) = scope; }} />
    );
}
}
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}

```

[Functional-component]

INDEX.JSX

```

{% raw %}
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    let calendarInstance;
    React.useEffect(() => {
        const proxy = this;
        document.onkeyup = (e) => {
            if (e.altKey && e.keyCode === 84 /* t */) {
                // press altt to focus the control.

proxy.calendarInstance.element.querySelector('table').focus();
            }
        };
    }, []);
    return (
        // specifies the tag for render the Calendar component
        <CalendarComponent id="calendar" ref={(scope) => { calendarInstance =
scope; }}/>);
}
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import { Calendar, CalendarComponent } from '@syncfusion/ej2-react-
calendars';
import * as React from "react";

```

```
import * as ReactDOM from "react-dom";
function App() {
  let calendarInstance: Calendar;
  React.useEffect(() => {
    const proxy = this;
    document.onkeyup = (e) => {
      if (e.altKey && e.keyCode === 84 /* t */) {
        // press altt to focus the control.
        proxy.calendarInstance.element.querySelector('table').focus();
      }
    };
  }, []);
  return (
    // specifies the tag for render the Calendar component
    <CalendarComponent id="calendar" ref={(scope) => { (calendarInstance
as Calendar | null) = scope; }} />
  );
}
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}
```

Ensuring accessibility

The Calendar component's accessibility levels are ensured through an [accessibility-checker](#) and [axe-core](#) software tools during automated testing.

The accessibility compliance of the Calendar component is shown in the following sample. Open the [sample](#) in a new window to evaluate the accessibility of the Calendar component with accessibility tools.

See also

- [Accessibility in Syncfusion React components](#)

Islamic calendar in React Calendar component

In addition to the Gregorian calendar, the Calendar control supports displaying the Islamic calendar (Hijri calendar). **Islamic calendar** or **Hijri calendar** is a **lunar calendar** consisting of 12 months in a year of 354 or 355 days. To know more about Islamic calendar, please refer this [wikipedia](#).

Also, it consists of all Gregorian calendar functionalities as like min and max date, week number, start day of the week, multi selection, enable RTL, start and depth view, localization, highlight and customize the specific dates.

By default, calendar mode is in **Gregorian**. You can enable the Islamic mode by setting the **calendarMode** as **Islamic**. Also, need to import and injecting the **Islamic** module into the Calendar using the **Inject** directive from **ej2-react-calendars** as shown below.

```
import { Islamic } from '@syncfusion/ej2-react-calendars';
```

By default, calendar mode is in **Gregorian**. You can enable the Islamic mode by setting the **calendarMode** as **Islamic** and injecting the **Islamic** module into the Calendar using the **Inject** directive.

The following example demonstrates how to display the Islamic Calendar (Hijri Calendar).

[Class-component]

INDEX.JSX

```

import { addClass } from '@syncfusion/ej2-base';
// import the calendarcomponent
import { CalendarComponent, Inject, Islamic } from '@syncfusion/ej2-react-
calendars';
import * as React from "react";
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    calendarMode = 'Islamic';
    constructor(props) {
        super(props);
        this.disableDate = this.disableDate.bind(this);
    }
    // initialize the calendar mode
    disableDate(args) {
        /*Date need to be disabled*/
        if (args.date.getDate() === 2 || args.date.getDate() === 10 ||
args.date.getDate() === 28) {
            args.isDisabled = true;
        }
        if (args.date.getDate() === 13) {
            let span;
            span = document.createElement('span');
            args.element.children[0].className += 'e-day sf-icon-cup
highlight';
            addClass([args.element], ['special', 'e-day', 'dinner']);
            args.element.setAttribute('data-val', ' Dinner !');
            args.element.appendChild(span);
        }
        if (args.date.getDate() === 23) {
            args.element.children[0].className += 'e-day sf-icon-start
highlight';
            let span;
            span = document.createElement('span');
            span.setAttribute('class', 'sf-icons-star highlight');
            // use the imported method to add the multiple classes to the
given element
            addClass([args.element], ['special', 'e-day', 'holiday']);
            args.element.setAttribute('data-val', ' Holiday !');
            args.element.appendChild(span);
        }
    }
    render() {
        return <CalendarComponent calendarMode={this.calendarMode}
renderDayCell={this.disableDate}><Inject
services={ [Islamic]} /></CalendarComponent>;
    }
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { addClass } from '@syncfusion/ej2-base';
// import the calendarcomponent

```



```

import { CalendarComponent, CalendarType, Inject, Islamic,
RenderDayCellEventArgs } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}>, {}> {
    private calendarMode: CalendarType = 'Islamic';
    constructor(props: any) {
        super(props);
        this.disableDate = this.disableDate.bind(this);
    }
    // initialize the calendar mode
    public disableDate(args: RenderDayCellEventArgs) {
        /*Date need to be disabled*/
        if ((args.date as Date).getDate() === 2 || (args.date as
Date).getDate() === 10 || (args.date as Date).getDate() === 28) {
            args.isDisabled = true;
        }
        if ((args.date as Date).getDate() === 13) {
            let span: HTMLElement;
            span = document.createElement('span');
            (args.element as HTMLElement).children[0].className += 'e-day
sf-icon-cup highlight';
            addClass([args.element as HTMLElement], ['special', 'e-day',
'dinner']);
            (args.element as HTMLElement).setAttribute('data-val', ' Dinner
!');
            (args.element as HTMLElement).appendChild(span);
        }
        if ((args.date as Date).getDate() === 23) {
            (args.element as HTMLElement).children[0].className += 'e-day
sf-icon-start highlight';
            let span: HTMLElement;
            span = document.createElement('span');
            span.setAttribute('class', 'sf-icons-star highlight');
            // use the imported method to add the multiple classes to the
given element
            addClass([args.element as HTMLElement], ['special', 'e-day',
'holiday']);
            (args.element as HTMLElement).setAttribute('data-val', ' Holiday
!');
            (args.element as HTMLElement).appendChild(span);
        }
    }
    public render() {
        return <CalendarComponent calendarMode={this.calendarMode}
renderDayCell={this.disableDate} ><Inject services=[[Islamic]]
/></CalendarComponent>
    }
};
ReactDOM.render(<App />, document.getElementById('element'));

```

[Functional-component]

INDEX.JSX

```

import { addClass } from '@syncfusion/ej2-base';

```

```
// import the calendarcomponent
import { CalendarComponent, Inject, Islamic } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from 'react-dom';
function App() {
    const calendarMode = 'Islamic';
    // initialize the calendar mode
    function disableDate(args) {
        /*Date need to be disabled*/
        if (args.date.getDate() === 2 || args.date.getDate() === 10 ||
args.date.getDate() === 28) {
            args.isDisabled = true;
        }
        if (args.date.getDate() === 13) {
            let span;
            span = document.createElement('span');
            args.element.children[0].className += 'e-day sf-icon-cup
highlight';
            addClass([args.element], ['special', 'e-day', 'dinner']);
            args.element.setAttribute('data-val', ' Dinner !');
            args.element.appendChild(span);
        }
        if (args.date.getDate() === 23) {
            args.element.children[0].className += 'e-day sf-icon-start
highlight';
            let span;
            span = document.createElement('span');
            span.setAttribute('class', 'sf-icons-star highlight');
            // use the imported method to add the multiple classes to the
given element
            addClass([args.element], ['special', 'e-day', 'holiday']);
            args.element.setAttribute('data-val', ' Holiday !');
            args.element.appendChild(span);
        }
    }
    return <CalendarComponent calendarMode={calendarMode}
renderDayCell={disableDate}><Inject
services={[Islamic]}></CalendarComponent>;
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import { addClass } from '@syncfusion/ej2-base';
// import the calendarcomponent
import { CalendarComponent, CalendarType, Inject, Islamic,
RenderDayCellEventArgs } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from 'react-dom';
function App() {
    const calendarMode: CalendarType = 'Islamic';
    // initialize the calendar mode
    function disableDate(args: RenderDayCellEventArgs) {
        /*Date need to be disabled*/
```

```

        if ((args.date as Date).getDate() === 2 || (args.date as
Date).getDate() === 10 || (args.date as Date).getDate() === 28) {
            args.isDisabled = true;
        }
        if ((args.date as Date).getDate() === 13) {
            let span: HTMLElement;
            span = document.createElement('span');
            (args.element as HTMLElement).children[0].className += 'e-day
sf-icon-cup highlight';
            addClass([args.element as HTMLElement], ['special', 'e-day',
'dinner']);
            (args.element as HTMLElement).setAttribute('data-val', ' Dinner
!');
            (args.element as HTMLElement).appendChild(span);
        }
        if ((args.date as Date).getDate() === 23) {
            (args.element as HTMLElement).children[0].className += 'e-day
sf-icon-start highlight';
            let span: HTMLElement;
            span = document.createElement('span');
            span.setAttribute('class', 'sf-icons-star highlight');
            // use the imported method to add the multiple classes to the
given element
            addClass([args.element as HTMLElement], ['special', 'e-day',
'holiday']);
            (args.element as HTMLElement).setAttribute('data-val', ' Holiday
!');
            (args.element as HTMLElement).appendChild(span);
        }
    }
    return <CalendarComponent calendarMode={calendarMode}
renderDayCell={disableDate} ><Inject services={[Islamic]}
/></CalendarComponent>
};
ReactDOM.render(<App />, document.getElementById('element'));

```

Style appearance in React Calendar component

The following content provides the exact CSS structure that can be used to modify the control's appearance based on the user preference.

Customizing the background color for the Calendar

Use the following CSS to customize the background color and border for the Calendar.

```

`css
/ To specify background color and border /
.e-calendar {
background-color: peachpuff;
border: 3px solid red;
}
`

```

Customizing the Calendar date elements on hovering

Use the following CSS to customize the date elements on hovering in the Calendar.

```
`css
/ To specify background color, color, and border /
.e-calendar .e-content td:hover span.e-day, .e-calendar .e-content td:focus span.e-day, .e-bigger.e-small
.e-calendar .e-content td:hover span.e-day, .e-bigger.e-small .e-calendar .e-content td:focus span.e-day
{
background-color: red;
border: 2px solid;
color: #212529;
}
`
```

Customizing the border of date cell grid

Use the following CSS to add the border to the date cell grid.

```
`css
/ To specify border /
.e-calendar .e-content span.e-day, .e-bigger.e-small .e-calendar .e-content span.e-day {
border: 1px solid;
}
`
```

Customizing the Calendar title

Use the following CSS to customize the Calendar title.

```
`css
/ To specify color and font size /
.e-calendar .e-header .e-title, .e-bigger.e-small .e-calendar .e-header .e-title {
color: black;
font-size: 20px;
}
`
```

Customizing the previous and next icon

Use the following CSS to customize the previous and next icon.

```
`css
/ To specify color and border /
.e-calendar .e-header span, .e-bigger.e-small .e-calendar .e-header span {
border: 1px solid;
}
```

```
color: chocolate;
}
`
```

Customizing the footer button

Use the following CSS to customize the footer button.

```
`css
/ To specify background color, color, and border-color /
.e-calendar .e-btn.e-today.e-flat.e-primary, .e-calendar .e-css.e-btn.e-today.e-flat.e-primary {
background-color: red;
border-color: black;
color: black;
}
`
```

Customizing the selected date cell grid

Use the following CSS to customize the selected date cell grid in Calendar.

```
`css
/ To specify background color and color /
.e-calendar .e-content td.e-focused-date.e-today span.e-day {
background-color: maroon;
color: #fff;
}
`
```

Customizing the content header in Calendar

Use the following CSS to customize the content header in Calendar.

```
`css
/ To specify background /
.e-calendar .e-content thead, .e-bigger.e-small .e-calendar .e-content thead {
background: aquamarine;
}
`
```

How To

Set clear button in calendar in React Calendar component

The following steps illustrate how to configure clear button in Calendar UI.

- On [created](#) event of Calendar add the required elements to have clear button. Here we have used div with Essential JS 2 Button component.
- Use the `e-footer` class to the div tag to act as the footer.
- And use the button to clear the selected date.
- Bind the required event handler to the button tags to update the value.

Below is the code example

INDEX.JSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
  onCreate() {
    const proxy = this;
    const clearBtn = document.createElement('button');
    clearBtn.setAttribute('class', 'e-btn e-clear e-flat');
    clearBtn.innerHTML = 'Clear';
    const footerElement = document.querySelector('.e-footer-container');
    footerElement.insertBefore(clearBtn, footerElement.childNodes[0]);
    document.querySelector(".e-clear").addEventListener("click", (e) => {
      proxy.value = null;
    });
  }
  render() {
    return (<CalendarComponent id="calendar"
    created={this.onCreate}/>);
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
  public onCreate(): void {
    const proxy: any = this;
    const clearBtn = document.createElement('button');
    clearBtn.setAttribute('class', 'e-btn e-clear e-flat');
    clearBtn.innerHTML = 'Clear';
    const footerElement: HTMLElement = document.querySelector('.e-
    footer-container') as HTMLElement;
    footerElement.insertBefore(clearBtn, footerElement.childNodes[0]);
    (document.querySelector(".e-clear") as
    HTMLElement).addEventListener("click", (e) => {
      proxy.value = null;
    });
  }
  public render() {
```

```

        return (
            <CalendarComponent id="calendar" created={this.onCreated} />
        );
    }
}
ReactDOM.render(<App />, document.getElementById('element'));

```

Show dates of other months in React Calendar component

The following example demonstrates how to show dates in other months.

The below styles show the Calendar's other month dates to visibility from its hidden state.

`css

```

.e-calendar .e-content tr.e-month-hide,
.e-calendar .e-content td.e-other-month>span.e-day {
display: block;
}
.e-calendar .e-content td.e-month-hide,
.e-calendar .e-content td.e-other-month {
pointer-events: auto;
touch-action: auto;
}
`

```

INDEX.JSX

```

// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
    render() {
        return <CalendarComponent id="calendar"/>;
    }
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

// import the calendarcomponent
import { CalendarComponent} from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
    public render() {
        return <CalendarComponent id="calendar" />
    }
};

```

```
ReactDOM.render(<App />, document.getElementById('element'));
```

Select a sequence of dates in calendar in React Calendar component

The following example demonstrates how to select the week dates of chosen date in the Calendar using [values](#) property, when [isMultiSelection](#) property is enabled. Methods of Moment.js is used in this sample for calculating the start and end of week from the selected date.

To parse, format and manipulating the date values, here we used momentum. You can install it by using below command

```
npm i moment
```

INDEX.JSX

```
{% raw %}
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
// import the calendar component
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as moment from "moment";
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
  calendarInstance;
  constructor(props) {
    super(props);
    this.onChange = this.onChange.bind(this);
    this.onweekChange = this.onweekChange.bind(this);
    this.onworkweekChange = this.onworkweekChange.bind(this);
  }
  onChange(args) {
    const startOfWeek = moment(args.value).startOf('week');
    const endOfWeek = moment(args.value).endOf('week');
    if (this.calendarInstance.element.classList.contains('workweek')) {
      this.getWeekArray(startOfWeek.day(1), endOfWeek.day(5));
    }
    else if (this.calendarInstance.element.classList.contains("week")) {
      this.getWeekArray(startOfWeek, endOfWeek);
    }
  }
  onworkweekChange() {
    if (this.calendarInstance.element.classList.contains('week')) {
      this.calendarInstance.element.classList.remove('week');
    }
    this.calendarInstance.element.classList.add('workweek');
  }
  onweekChange() {
    if (this.calendarInstance.element.classList.contains('workweek')) {
      this.calendarInstance.element.classList.remove('workweek');
    }
    this.calendarInstance.element.classList.add('week');
  }
  getWeekArray(startOfWeek, endOfWeek) {
    const days = [];
    let day = startOfWeek;
    while (day <= endOfWeek) {
      days.push(day.toDate());
    }
  }
}
```



```

        day = day.clone().add(1, 'd');
    }
    this.calendarInstance.values = days;
}
render() {
    return (<div id="app">
        <div className="wrap">
            <CalendarComponent id="calendar" isMultiSelection={true}
change={this.onChange} ref={ (element) => { this.calendarInstance = element;
}}/>
        </div>
        <div id="btncontainer" className="e-btn-group e-vertical">
            <ButtonComponent onClick={this.onweekChange} id="week"
className="e-btn"> Week </ButtonComponent>
            <ButtonComponent onClick={this.onworkweekChange} id="workweek"
className="e-btn"> Work Week </ButtonComponent>
        </div>
    </div>);
}
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
// import the calendarcomponent
import { Calendar, CalendarComponent, ChangedEventArgs } from
 '@syncfusion/ej2-react-calendars';
import * as moment from "moment";
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
private calendarInstance: Calendar;
constructor(props: any) {
    super(props);
    this.onChange = this.onChange.bind(this);
    this.onweekChange = this.onweekChange.bind(this);
    this.onworkweekChange = this.onworkweekChange.bind(this);
}
public onChange(args: ChangedEventArgs) {
    const startOfWeek: any = moment(args.value).startOf('week');
    const endOfWeek: any = moment(args.value).endOf('week');
    if (this.calendarInstance.element.classList.contains('workweek')) {
        this.getWeekArray(startOfWeek.day(1), endOfWeek.day(5));
    } else if (this.calendarInstance.element.classList.contains("week")) {
        this.getWeekArray(startOfWeek, endOfWeek);
    }
}
public onworkweekChange(): void {
    if (this.calendarInstance.element.classList.contains('week')) {
        this.calendarInstance.element.classList.remove('week');
    }
    this.calendarInstance.element.classList.add('workweek');
}
}

```

```

public onweekChange(): void{
    if (this.calendarInstance.element.classList.contains('workweek')) {
        this.calendarInstance.element.classList.remove('workweek')
    }
    this.calendarInstance.element.classList.add('week');
}
public getWeekArray(startOfWeek: any ,endOfWeek: any) {
    const days = [];
    let day = startOfWeek;
    while (day <= endOfWeek) {
        days.push(day.toDate());
        day = day.clone().add(1, 'd');
    }
    this.calendarInstance.values = days;
}
public render() {
    return (
        <div id="app">
            <div className="wrap">
                <CalendarComponent id="calendar" isMultiSelection={true}
change={this.onChange} ref={ (element) => {(this.calendarInstance as Calendar
| null) = element }} />
            </div>
            <div id="btncontainer" className="e-btn-group e-vertical">
                <ButtonComponent onClick={this.onweekChange} id="week"
className="e-btn"> Week </ButtonComponent>
                <ButtonComponent onClick={this.onworkweekChange} id="workweek"
className="e-btn"> Work Week </ButtonComponent>
            </div>
        </div>
    );
}
}
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}

```

Skip a month in calendar in React Calendar component

The following example demonstrates how to skip a month in a Calendar while clicking the previous and next icon. Here we have used the [navigated](#) event to skip a month using [NavigateTo](#) method.

INDEX.JSX

```

{% raw %}
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    calendarInstance;
    // skips a month while clicking previous and next icon in month view.
    onNavigate(args) {
        let date = 0;
        if (args.event.currentTarget.classList.contains('e-next')) {
            // increment the month while clicking the next icon
            date = args.date.setMonth(args.date.getMonth() + 1);
        }
    }
}

```

```

        if (args.event.currentTarget.classList.contains('e-prev')) {
            // decrement the month while clicking the previous icon
            date = args.date.setMonth(args.date.getMonth() - 1);
        }
        if (args.view === 'month') {
            this.calendarInstance.navigateTo('Month', new Date(date));
        }
    }
    render() {
        return (<CalendarComponent navigated={this.onNavigate =
this.onNavigate.bind(this)} ref={(scope) => { this.calendarInstance = scope;
}}/>);
    }
}
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
// import the calendarcomponent
import { Calendar, CalendarComponent, NavigatedEventArgs } from
'@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    public calendarInstance: Calendar;
    // skips a month while clicking previous and next icon in month view.
    public onNavigate(args: NavigatedEventArgs) {
        let date: number = 0;
        if (((args.event as KeyboardEvent | MouseEvent |
Event).currentTarget as HTMLElement).classList.contains('e-next')) {
            // increment the month while clicking the next icon
            date = (args.date as Date).setMonth((args.date as
Date).getMonth() + 1);
        }
        if (((args.event as KeyboardEvent | MouseEvent |
Event).currentTarget as HTMLElement).classList.contains('e-prev')) {
            // decrement the month while clicking the previous icon
            date = (args.date as Date).setMonth((args.date as
Date).getMonth() - 1);
        }
        if (args.view === 'month') {
            this.calendarInstance.navigateTo('Month', new Date(date));
        }
    }
    public render() {
        return (
            <CalendarComponent navigated={this.onNavigate =
this.onNavigate.bind(this)} ref={(scope) => {(this.calendarInstance as
Calendar | null) = scope}} />
        );
    }
}
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}

```

Render the calendar with week numbers in React Calendar component

You can enable the `weekNumber` in Calendar by using the [weekNumber](#)

property.

INDEX.JSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
  render() {
    return <CalendarComponent id="calendar" weekNumber={true}/>;
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
  public render() {
    return <CalendarComponent id="calendar" weekNumber={true} />
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

Change the first day of week in React Calendar component

The Calendar provides an option to change the first day of the week by using the [firstDayOfWeek](#) property. Day of the week starts from 0(Sunday) to 6(Saturday).

By default, the first day of week is based on culture specific.

The following example demonstrates the Calendar with `Tuesday` as first day of the week.

INDEX.JSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
  weekstart = 2;
  render() {
    return <CalendarComponent id="calendar"
firstDayOfWeek={this.weekstart}/>;
  }
};
;
```

```
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}>, {}> {
    private weekstart:number = 2;
    public render() {
        return <CalendarComponent id="calendar"
firstDayOfWeek={this.weekstart} />
    }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

Customize the calendar day header in React Calendar component

You can change the format of the day that to be displayed in header using [dayHeaderFormat](#) property. By default, the format is **Short**.

You can find the possible formats on below.

| Name | Description |
|-------------|--|
| ----- ----- | |
| Short | Sets the short format of day name (like Su) in day header. |
| Narrow | Sets the single character of day name (like S) in day header. |
| Abbreviated | Sets the min format of day name (like Sun) in day header. |
| Wide | Sets the long format of day name (like Sunday) in day header. |

INDEX.JSX

```
{% raw %}
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
    calendarObj;
    floatLabelObj;
    floatData;
    fields;
    value = 'Short';
    constructor(props) {
        super(props);
        this.floatData = [
            { Id: 'Short', Label: 'Short' },
            { Id: 'Narrow', Label: 'Narrow' },
            { Id: 'Abbreviated', Label: 'Abbreviated' },
            { Id: 'Wide', Label: 'Wide' }
        ];
    }
};
```

```

        this.fields = { text: 'Label', value: 'Id' };
    }
    formatHandler(args) {
        this.calendarObj.dayHeaderFormat = args.value;
    }
    render() {
        return (<div id="container">
            <div id="calendar">
                <CalendarComponent ref={ (calendar) => this.calendarObj =
calendar} dayHeaderFormat="Short"/>
            </div>
            <div id="format">
                <label class="custom-input-label">Header Format
Types</label>
                <DropDownListComponent id="select" value={this.value}
dataSource={this.floatData} ref={ (dropdownlist) => { this.floatLabelObj =
dropdownlist; }} fields={this.fields} change={this.formatHandler.bind(this)}
placeholder="Select float type"/>
            </div>
        </div>);
    }
}
;
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
// import the calendarcomponent
import { CalendarComponent } from '@syncfusion/ej2-react-calendars';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
    public calendarObj: CalendarComponent;
    public floatLabelObj: DropDownListComponent;
    private floatData: { [key: string]: Object }[];
    private fields: object;
    private value: string = 'Short';
    constructor(props: {}) {
        super(props);
        this.floatData = [
            { Id: 'Short', Label: 'Short' },
            { Id: 'Narrow', Label: 'Narrow' },
            { Id: 'Abbreviated', Label: 'Abbreviated' },
            { Id: 'Wide', Label: 'Wide' }
        ];
        this.fields = { text: 'Label', value: 'Id' };
    }

    private formatHandler(args: any): void {
        this.calendarObj.dayHeaderFormat = args.value;
    }

    public render(): JSX.Element {
        return (
            <div id="container">

```

```

        <div id="calendar">
            <CalendarComponent ref={(calendar) => this.calendarObj =
calendar} dayHeaderFormat="Short" />
        </div>
        <div id="format">
            <label class="custom-input-label">Header Format
Types</label>
            <DropDownListComponent id="select" value = {this.value}
dataSource={this.floatData} ref={(dropdownlist) => { this.floatLabelObj =
dropdownlist }} fields={this.fields} change={this.formatHandler.bind(this)}
placeholder="Select float type" />
        </div>
    </div>
    );
}
};
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}

```

Card

Getting Started

This section explains how to create a simple **Card** using Styles, and how to configure the structure for the header section, Horizontal, action buttons, content section.

Dependencies

The Card Component is pure CSS component so no specific dependencies to render the card.

Setup for Local Development

You can use [create-react-app](#) to setup the applications. To install `create-react-app` run the following command.

`

```
npm install -g create-react-app
```

`

- To setup basic `React` sample use following commands.

using TSX

`

```
create-react-app quickstart --scripts-version=react-scripts-ts
```

```
cd quickstart
```

`

using JSX

`

```
create-react-app quickstart
```

```
cd quickstart
```

Adding Syncfusion packages

All the available Essential JS 2 packages are published in [npmjs.com](https://www.npmjs.com) public registry.

Install the below required dependency package in order to use the **Card** component in your application.

```
`bash
```

```
npm install @syncfusion/ej2-layouts --save
```

- The Card CSS files are available in the **ej2-layouts** package folder. This can be referenced in your application using the following code.

```
[src/styles/styles.css]
```

```
`css
```

```
@import '../node_modules/@syncfusion/ej2-layouts/styles/material.css';
```

Adding a simple Card

- Add the HTML **div** element with **e-card** class into your **index.html**.

```
[src/index.html]
```

```
<div className = "e-card">
```

```
Sample Card
```

```
</div>
```

Adding a header to the card

You can create cards with a header in a specific structure. For adding header you need to create **div** element and add **e-card-header** class.

- You can include heading inside the card header by adding an **div** element with **e-card-header-caption** class, and also content will be added by adding element with **e-card-content**. For detailed information, refer to the [Header and Content](#).

```
<div class = "e-card"> --> Root Element
```

```
<div class="e-card-header"> --> Root Header Element
```

```
<div class="e-card-header-caption"> --> Root Heading Element
```

```
<div class="e-card-header-title"></div> --> Heading Title Element
```



```

</div>
<div class="e-card-content"></div>    --> Card content Element
</div>
</div>
`

```

- Now, run the application in the browser using the following command.

```

`
npm start
`

```

Output will be as follows:

[Class-component]

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component {
  render() {
    return (
      <div>
        <div className="e-card" id="basic">
          <div className="e-card-header">
            <div className="e-card-header-caption">
              <div className="e-card-title">Advanced UWP</div>
            </div>
          </div>
          <div className="e-card-content">
            Communicating with Windows 10 and Other Apps, the second in a
            five-part series written by Succinctly series
            author Matteo Pagani. To download the complete white paper, and
            other papers in the series, visit
            the White Paper section of Syncfusion's Technology Resource
            Portal.
          </div>
        </div>
      </div>);
  }
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component<{}, {}> {
  public render() {
    return (
      <div>
        <div className="e-card" id="basic">

```

```

        <div className="e-card-header">
            <div className="e-card-header-caption">
                <div className="e-card-title">Advanced UWP</div>
            </div>
        </div>
        <div className="e-card-content">
            Communicating with Windows 10 and Other Apps, the second in a
            five-part series written by Succinctly series
            author Matteo Pagani. To download the complete white paper, and
            other papers in the series, visit
            the White Paper section of Syncfusion's Technology Resource
            Portal.
        </div>
    </div>
</div>
);
}
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
    <title>Syncfusion React Card Sample</title>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta name="description" content="Essential JS 2 for React Components"
/>
    <meta name="author" content="Syncfusion" />
    <link href="https://cdn.syncfusion.com/ej2/20.3.56/ej2-
layouts/styles/material.css" rel="stylesheet" />
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
    <script src="systemjs.config.js"></script>
</head>
<body>
    <div id='element'>
        <div id='loader'>Loading....</div>
    </div>
</body>
</html>

```

[Functional-component]

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
function ReactApp() {
    return <div>
        <div className="e-card" id="basic">
            <div className="e-card-header">
                <div className="e-card-header-caption">

```

```

        <div className="e-card-title">Advanced UWP</div>
      </div>
    </div>
    <div className="e-card-content">
      Communicating with Windows 10 and Other Apps, the second in a
      five-part series written by Succinctly series
      author Matteo Pagani. To download the complete white paper, and
      other papers in the series, visit
      the White Paper section of Syncfusion's Technology Resource
      Portal.
    </div>
  </div>
</div>);
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
function ReactApp () {
  return (
    <div>
      <div className="e-card" id="basic">
        <div className="e-card-header">
          <div className="e-card-header-caption">
            <div className="e-card-title">Advanced UWP</div>
          </div>
        </div>
        <div className="e-card-content">
          Communicating with Windows 10 and Other Apps, the second in a
          five-part series written by Succinctly series
          author Matteo Pagani. To download the complete white paper, and
          other papers in the series, visit
          the White Paper section of Syncfusion's Technology Resource
          Portal.
        </div>
      </div>
    </div>
  );
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
  <title>Syncfusion React Card Sample</title>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <meta name="description" content="Essential JS 2 for React Components"
/>
  <meta name="author" content="Syncfusion" />
  <link href="https://cdn.syncfusion.com/ej2/20.3.56/ej2-
layouts/styles/material.css" rel="stylesheet" />

```

```
<script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
  <script src="systemjs.config.js"></script>
</head>
<body>
  <div id='element'>
    <div id='loader'>Loading....</div>
  </div>
</body>
</html>
```

See Also

- [How to add a header and content](#)

Header content in React Card component

Header

The Card can be created with header title, sub title and images. For adding header you need to create `div` element with the class `e-card-header` added.

Card provides below elements and corresponding class definitions to include header.

Elements | Description

`e-card-header-caption` | To group the title and subtitle within the header which acts as wrapper.

`e-card-header-title` | Main title text with in the header.

`e-card-sub-title` | A sub-title within the header.

`e-card-header-image` | To include heading image within the header.

`e-card-corner` | To add rounded corner for the image.

Title and Subtitle

For adding header to the Card , you need to create wrapper `div` element with `e-card-header-caption` class.

- Place the `div` element with `e-card-header-title` class inside the header caption for adding main title.
- Place the `div` element with `e-card-sub-title` class inside the header caption element for adding sub-title.

Image

Card header has an option for adding images in the header. It is aligned with either before or after the header based on the HTML element positioned in the header structure.

- The header image can be added by creating a `div` element with `e-card-header-image` class which can be placed before or after the header caption wrapper element.

[Class-component]

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component {
  render() {
    return (
      <div>
        <div className="e-card">
          <div className="e-card-header">
            <div className="e-card-header-image football e-card-corner"/>
            <div className="e-card-header-caption">
              <div className="e-card-header-title"> Laura Callahan</div>
              <div className="e-card-sub-title">Sales Coordinator and
Representative</div>
            </div>
          </div>
        </div>
        <div className="e-card">
          <div className="e-card-header">
            <div className="e-card-header-caption">
              <div className="e-card-header-title"> Laura Callahan</div>
              <div className="e-card-sub-title">Sales Coordinator and
Representative</div>
            </div>
            <div className="e-card-header-image football"/>
          </div>
        </div>
      </div>);
  }
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component<{}, {}> {
  public render() {
    return (
      <div>
        <div className="e-card">
          <div className="e-card-header">
            <div className="e-card-header-image football e-card-corner" />
            <div className="e-card-header-caption">
              <div className="e-card-header-title"> Laura Callahan</div>
              <div className="e-card-sub-title">Sales Coordinator and
Representative</div>
            </div>
          </div>
        </div>
        <div className="e-card">
          <div className="e-card-header">
            <div className="e-card-header-caption">
              <div className="e-card-header-title"> Laura Callahan</div>
              <div className="e-card-sub-title">Sales Coordinator and
Representative</div>
            </div>
            <div className="e-card-header-image football"/>
          </div>
        </div>
      </div>);
  }
}
```

```

        </div>
        <div className="e-card-header-image football" />
      </div>
    </div>
  </div>
);
}
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
  <title>Syncfusion React Card Sample</title>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <meta name="description" content="Essential JS 2 for React Components"
/>
  <meta name="author" content="Syncfusion" />
  <link href="https://cdn.syncfusion.com/ej2/20.3.56/ej2-
layouts/styles/material.css" rel="stylesheet" />
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
  <script src="systemjs.config.js"></script>
<style>
  .e-card .e-card-header .e-card-header-image {
    background-image: url('../football.png');
  }
  .map {
    background-image: url('../map.png');
    height: 200px;
  }
  .e-card {
    width: 300px
  }
  .e-card:last-child {
    margin-top: 30px;
  }
</style>
</head>
<body>
  <div id='element'>
    <div id='loader'>Loading....</div>
  </div>
</body>
</html>

```

[Functional-component]

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";

```

```
function ReactApp() {
  return (<div>
    <div className="e-card">
      <div className="e-card-header">
        <div className="e-card-header-image football e-card-corner"/>
        <div className="e-card-header-caption">
          <div className="e-card-header-title"> Laura Callahan</div>
          <div className="e-card-sub-title">Sales Coordinator and
Representative</div>
        </div>
      </div>
    </div>
    <div className="e-card">
      <div className="e-card-header">
        <div className="e-card-header-caption">
          <div className="e-card-header-title"> Laura Callahan</div>
          <div className="e-card-sub-title">Sales Coordinator and
Representative</div>
        </div>
        <div className="e-card-header-image football"/>
      </div>
    </div>
  </div>);
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
function ReactApp () {
  return (
    <div>
      <div className="e-card">
        <div className="e-card-header">
          <div className="e-card-header-image football e-card-corner" />
          <div className="e-card-header-caption">
            <div className="e-card-header-title"> Laura Callahan</div>
            <div className="e-card-sub-title">Sales Coordinator and
Representative</div>
          </div>
        </div>
      </div>
      <div className="e-card">
        <div className="e-card-header">
          <div className="e-card-header-caption">
            <div className="e-card-header-title"> Laura Callahan</div>
            <div className="e-card-sub-title">Sales Coordinator and
Representative</div>
          </div>
          <div className="e-card-header-image football" />
        </div>
      </div>
    </div>
  );
}
```

```
ReactDOM.render(<ReactApp />, document.getElementById("element"));
```

INDEX.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Syncfusion React Card Sample</title>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <meta name="description" content="Essential JS 2 for React Components"
/>
  <meta name="author" content="Syncfusion" />
  <link href="https://cdn.syncfusion.com/ej2/20.3.56/ej2-
layouts/styles/material.css" rel="stylesheet" />
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
  <script src="systemjs.config.js"></script>
<style>
  .e-card .e-card-header .e-card-header-image {
    background-image: url('../football.png');
  }
  .map {
    background-image: url('../map.png');
    height: 200px;
  }
  .e-card {
    width: 300px
  }
  .e-card:last-child {
    margin-top: 30px;
  }
</style>
</head>
<body>
  <div id='element'>
    <div id='loader'>Loading....</div>
  </div>
</body>
</html>
```

Content

Content in Card holds texts, images, links and all possible HTML elements. Its adaptable within the Card root element.

- Create a `div` element with the class `e-card-content`.
- Place content `div` element in the Card root element or within any Card inner elements.

[Class-component]

INDEX.JSX

```
import * as React from "react";
```



```
import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component {
  render() {
    return (<div>
      <div className="e-card">
        <div className="e-card-header">
          <div className="e-card-header-image football"/>
          <div className="e-card-header-caption">
            <div className="e-card-header-title"> Laura Callahan</div>
            <div className="e-card-sub-title">Sales Coordinator and
Representative</div>
          </div>
        </div>
        <div className="e-card-content">
          Laura received a BA in psychology from the University of
Washington. She has also completed a course in business French. She reads
and writes French.
        </div>
      </div>
    </div>);
  }
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component<{}, {}> {
  public render() {
    return (
      <div>
        <div className="e-card">
          <div className="e-card-header">
            <div className="e-card-header-image football" />
            <div className="e-card-header-caption">
              <div className="e-card-header-title"> Laura Callahan</div>
              <div className="e-card-sub-title">Sales Coordinator and
Representative</div>
            </div>
          </div>
          <div className="e-card-content">
            Laura received a BA in psychology from the University of
Washington. She has also completed a course in business French. She reads
and writes French.
          </div>
        </div>
      </div>
    );
  }
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));
```

INDEX.HTML

```
<!DOCTYPE html>
```

```

<html lang="en">
<head>
  <title>Syncfusion React Card Sample</title>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <meta name="description" content="Essential JS 2 for React Components"
/>
  <meta name="author" content="Syncfusion" />
  <link href="https://cdn.syncfusion.com/ej2/20.3.56/ej2-
layouts/styles/material.css" rel="stylesheet" />
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
  <script src="systemjs.config.js"></script>
<style>
  .e-card .e-card-header .e-card-header-image {
    background-image: url('./football.png');
  }
  .map {
    background-image: url('./map.png');
    height: 200px;
  }
  .e-card {
    width: 300px
  }
  .e-card:last-child {
    margin-top: 30px;
  }
</style>
</head>
<body>
  <div id='element'>
    <div id='loader'>Loading....</div>
  </div>
</body>
</html>

```

[Functional-component]

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
function ReactApp() {
  return (<div>
    <div className="e-card">
      <div className="e-card-header">
        <div className="e-card-header-image football"/>
        <div className="e-card-header-caption">
          <div className="e-card-header-title"> Laura Callahan</div>
          <div className="e-card-sub-title">Sales Coordinator and
Representative</div>
        </div>
      </div>
      <div className="e-card-content">

```

```

        Laura received a BA in psychology from the University of
        Washington. She has also completed a course in business French. She reads
        and writes French.
    </div>
</div>
</div>);
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
function ReactApp () {
    return (
        <div>
            <div className="e-card">
                <div className="e-card-header">
                    <div className="e-card-header-image football" />
                    <div className="e-card-header-caption">
                        <div className="e-card-header-title"> Laura Callahan</div>
                        <div className="e-card-sub-title">Sales Coordinator and
Representative</div>
                    </div>
                </div>
                <div className="e-card-content">
                    Laura received a BA in psychology from the University of
                    Washington. She has also completed a course in business French. She reads
                    and writes French.
                </div>
            </div>
        </div>
    );
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
    <title>Syncfusion React Card Sample</title>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta name="description" content="Essential JS 2 for React Components"
/>
    <meta name="author" content="Syncfusion" />
    <link href="https://cdn.syncfusion.com/ej2/20.3.56/ej2-
layouts/styles/material.css" rel="stylesheet" />
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
    <script src="systemjs.config.js"></script>
<style>
    .e-card .e-card-header .e-card-header-image {
        background-image: url('../football.png');
    }

```

```

    }
    .map {
      background-image: url('./map.png');
      height: 200px;
    }
    .e-card {
      width: 300px
    }
    .e-card:last-child {
      margin-top: 30px;
    }
  </style>
</head>
<body>
  <div id='element'>
    <div id='loader'>Loading....</div>
  </div>
</body>
</html>

```

Card image in React Card component

Images

The Card supports to include images within the elements, you can add image as direct element anywhere inside card root by adding the `e-card-image` class to `div` element. Using the class defined, you can write CSS styles to load images to that element.

By default, card images occupies full width of its parent element.

,

```

<div className = "e-card">
  <div className="e-card-image">
  </div>
</div>

```

,

Title

Card image is supported to include a title or caption for the image. By default, Title is placed over the image on left-bottom position with overlay.

,

```

<div className = "e-card">
  <div className="e-card-image">
    <div className="e-card-title"></div>
  </div>
</div>

```

,

[Class-component]**INDEX.JSX**

```
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component {
  render() {
    return (<div className="e-card">
      <div className="e-card-image">
        <div className="e-card-title">JavaScript </div>
      </div>
      <div className="e-card-content">
        JavaScript Succinctly was written to give readers an accurate,
        concise examination of JavaScript objects and their supporting nuances, such
        as complex values, primitive values, scope, inheritance, the head object,
        and more. </div>
      </div>);
  }
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component<{}, {}> {
  public render() {
    return (
      <div className="e-card">
        <div className="e-card-image">
          <div className="e-card-title">JavaScript </div>
        </div>
        <div className="e-card-content">
          JavaScript Succinctly was written to give readers an accurate,
          concise examination of JavaScript objects and their supporting nuances, such
          as complex values, primitive values, scope, inheritance, the head object,
          and more. </div>
        </div>
      );
    );
  }
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));
```

INDEX.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Syncfusion React Card Sample</title>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <meta name="description" content="Essential JS 2 for React Components" />
  <meta name="author" content="Syncfusion" />
</head>
```

```

<link href="https://cdn.syncfusion.com/ej2/20.3.56/ej2-
layouts/styles/material.css" rel="stylesheet" />
<script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
<script src="systemjs.config.js"></script>
<style>
.e-card-image {
    background: url('./sample.jpg');
    height: 160px;
    ;
}
.e-card {
    width: 200px;
    margin: auto;
}
</style>
</head>
<body>
<div id='element'>
    <div id='loader'>Loading....</div>
</div>
</body>
</html>

```

[Functional-component]

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
function ReactApp() {
    return (<div className="e-card">
        <div className="e-card-image">
            <div className="e-card-title">JavaScript </div>
        </div>
        <div className="e-card-content">
            JavaScript Succinctly was written to give readers an accurate,
            concise examination of JavaScript objects and their supporting nuances, such
            as complex values, primitive values, scope, inheritance, the head object,
            and more. </div>
        </div>);
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
function ReactApp () {
    return (
        <div className="e-card">
            <div className="e-card-image">
                <div className="e-card-title">JavaScript </div>
            </div>
            <div className="e-card-content">

```

```

        JavaScript Succinctly was written to give readers an accurate,
        concise examination of JavaScript objects and their supporting nuances, such
        as complex values, primitive values, scope, inheritance, the head object,
        and more. </div>
    </div>
    );
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
    <title>Syncfusion React Card Sample</title>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta name="description" content="Essential JS 2 for React Components"
/>
    <meta name="author" content="Syncfusion" />
    <link href="https://cdn.syncfusion.com/ej2/20.3.56/ej2-
layouts/styles/material.css" rel="stylesheet" />
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
    <script src="systemjs.config.js"></script>
</style>
    .e-card-image {
        background: url('./sample.jpg');
        height: 160px;
        ;
    }
    .e-card {
        width: 200px;
        margin: auto;
    }
</style>
</head>
<body>
    <div id='element'>
        <div id='loader'>Loading....</div>
    </div>
</body>
</html>

```

Divider

Divider used to separate the elements inside the card. You can add divider inside the card elements to separate it.

- Place the `div` element with `e-card-separator` class inside the card element for adding a divider.

[Class-component]

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component {
  render() {
    return (
      <div>
        <div className="e-card" id="basic">
          <div className="e-card-title">Explore Cities</div>
          <div className="e-card-separator"/>
          <div className="e-card-content">
            Sydney is a city on the east coast of Australia. Sydney is the
            capital city of New South Wales. About four million people live in Sydney
            which makes it the biggest city in Oceania.
          </div>
          <div className="e-card-separator"/>
          <div className="e-card-content">
            New York City has been described as the cultural, financial, and
            media capital of the world, and exerts a significant impact upon commerce
            and etc.
          </div>
          <div className="e-card-separator"/>
          <div className="e-card-content">
            Malaysia is one of the Southeast Asian countries, on a peninsula
            of the Asian continent, to a certain extent; it can be recognized as part of
            the Asian continent.
          </div>
        </div>
      </div>);
  }
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component<{}, {}> {
  public render() {
    return (
      <div>
        <div className="e-card" id="basic">
          <div className="e-card-title">Explore Cities</div>
          <div className="e-card-separator" />
          <div className="e-card-content">
            Sydney is a city on the east coast of Australia. Sydney is the
            capital city of New South Wales. About four million people live in Sydney
            which makes it the biggest city in Oceania.
          </div>
          <div className="e-card-separator" />
          <div className="e-card-content">
            New York City has been described as the cultural, financial, and
            media capital of the world, and exerts a significant impact upon commerce
            and etc.
          </div>
          <div className="e-card-separator" />
          <div className="e-card-content">

```



```

        Malaysia is one of the Southeast Asian countries, on a peninsula
of the Asian continent, to a certain extent; it can be recognized as part of
the Asian continent.
    </div>
</div>
</div>
);
}
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
    <title>Syncfusion React Card Sample</title>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta name="description" content="Essential JS 2 for React Components"
/>
    <meta name="author" content="Syncfusion" />
    <link href="https://cdn.syncfusion.com/ej2/20.3.56/ej2-
layouts/styles/material.css" rel="stylesheet" />
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
    <script src="systemjs.config.js"></script>
</head>
<body>
    <div id='element'>
        <div id='loader'>Loading....</div>
    </div>
</body>
</html>

```

[Functional-component]

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
function ReactApp() {
    return <div>
        <div className="e-card" id="basic">
            <div className="e-card-title">Explore Cities</div>
            <div className="e-card-separator"/>
            <div className="e-card-content">
                Sydney is a city on the east coast of Australia. Sydney is the
capital city of New South Wales. About four million people live in Sydney
which makes it the biggest city in Oceania.
            </div>
            <div className="e-card-separator"/>
            <div className="e-card-content">

```

```

        New York City has been described as the cultural, financial, and
media capital of the world, and exerts a significant impact upon commerce
and etc.
    </div>
    <div className="e-card-separator"/>
    <div className="e-card-content">
        Malaysia is one of the Southeast Asian countries, on a peninsula
of the Asian continent, to a certain extent; it can be recognized as part of
the Asian continent.
    </div>
</div>
</div>);
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
function ReactApp () {
    return (
        <div>
            <div className="e-card" id="basic">
                <div className="e-card-title">Explore Cities</div>
                <div className="e-card-separator" />
                <div className="e-card-content">
                    Sydney is a city on the east coast of Australia. Sydney is the
capital city of New South Wales. About four million people live in Sydney
which makes it the biggest city in Oceania.
                </div>
                <div className="e-card-separator" />
                <div className="e-card-content">
                    New York City has been described as the cultural, financial, and
media capital of the world, and exerts a significant impact upon commerce
and etc.
                </div>
                <div className="e-card-separator" />
                <div className="e-card-content">
                    Malaysia is one of the Southeast Asian countries, on a peninsula
of the Asian continent, to a certain extent; it can be recognized as part of
the Asian continent.
                </div>
            </div>
        </div>
    );
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
    <title>Syncfusion React Card Sample</title>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

```

```

<meta name="description" content="Essential JS 2 for React Components"
/>
<meta name="author" content="Syncfusion" />
<link href="https://cdn.syncfusion.com/ej2/20.3.56/ej2-
layouts/styles/material.css" rel="stylesheet" />
<script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
<script src="systemjs.config.js"></script>
</head>
<body>
  <div id='element'>
    <div id='loader'>Loading....</div>
  </div>
</body>
</html>

```

See Also

- [How to customize the card image title position](#)

Action buttons in React Card component

You can include Action buttons within the Card and customize them. Action button is a `div` element with `e-card-actions` class followed by button tag or anchor tag within the card root element.

- For adding action buttons you can create button or anchor tag with `e-card-btn` class within the card action element.

```

,
<div className = "e-card">
<div className="e-card-actions">
<button className="e-card-btn"></button>
<a href="#"></a>
</div>
</div>
,

```

Vertical

By default, action buttons positioned in horizontal alignment , and also it can be aligned to show in vertical alignment by adding `e-card-vertical` class.

```

,
<div className = "e-card">
<div className="e-card-actions e-card-vertical">
<button className="e-card-btn">More</button>

```

```
<a href="#">Share</a>
```

```
</div>
```

```
</div>
```

```
,
```

[Class-component]

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component {
  render() {
    return (
      <div>
        <div id="sample">
          <div className="e-card">
            <div className="e-card-header-title">Eiffel Tower</div>
            <div className="e-card-content">
              The Eiffel Tower is acknowledged as the universal symbol of
Paris and France.
            </div>
            <div className="e-card-actions">
              <button className="e-card-btn">
                <img src='./fav.png' title="Bookmark"/>
              </button>
              <button className="e-card-btn">
                <img src='./like.png' title="Like"/>
              </button>
              <button className="e-card-btn">
                <img src='./share.png' title="Share"/>
              </button>
            </div>
          </div>
          <div id="sample">
            <div className="e-card">
              <div className="e-card-header-title">Eiffel Tower</div>
              <div className="e-card-content">
                The Eiffel Tower is acknowledged as the universal symbol of
Paris and France.
              </div>
              <div className="e-card-actions e-card-vertical">
                <button className="e-card-btn">More</button>
                <a href="#">Share</a>
              </div>
            </div>
          </div>
        </div>);
  }
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));
```

INDEX.TSX

```
import * as React from "react";
```

```

import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component<{}, {}> {
  public render() {
    return (
      <div>
        <div id="sample">
          <div className="e-card">
            <div className="e-card-header-title">Eiffel Tower</div>
            <div className="e-card-content">
              The Eiffel Tower is acknowledged as the universal symbol of
Paris and France.
            </div>
            <div className="e-card-actions">
              <button className="e-card-btn">
                <img src='./fav.png' title="Bookmark" />
              </button>
              <button className="e-card-btn">
                <img src='./like.png' title="Like" />
              </button>
              <button className="e-card-btn">
                <img src='./share.png' title="Share" />
              </button>
            </div>
          </div>
        </div>
        <div id="sample">
          <div className="e-card">
            <div className="e-card-header-title">Eiffel Tower</div>
            <div className="e-card-content">
              The Eiffel Tower is acknowledged as the universal symbol of
Paris and France.
            </div>
            <div className="e-card-actions e-card-vertical">
              <button className="e-card-btn">More</button>
              <a href="#">Share</a>
            </div>
          </div>
        </div>
      </div>
    );
  }
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
  <title>Syncfusion React Card Sample</title>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <meta name="description" content="Essential JS 2 for React Components"
/>
  <meta name="author" content="Syncfusion" />

```

```

<link href="https://cdn.syncfusion.com/ej2/20.3.56/ej2-
layouts/styles/material.css" rel="stylesheet" />
<link href="index.css" rel="stylesheet" />
<script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
<script src="systemjs.config.js"></script>
</head>
<body>
<div id='element'>
<div id='loader'>Loading....</div>
</div>
</body>
</html>

```

[Functional-component]

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
function ReactApp() {
  return (
    <div>
      <div id="sample">
        <div className="e-card">
          <div className="e-card-header-title">Eiffel Tower</div>
          <div className="e-card-content">
            The Eiffel Tower is acknowledged as the universal symbol of
            Paris and France.
          </div>
          <div className="e-card-actions">
            <button className="e-card-btn">
              <img src='./fav.png' title="Bookmark"/>
            </button>
            <button className="e-card-btn">
              <img src='./like.png' title="Like"/>
            </button>
            <button className="e-card-btn">
              <img src='./share.png' title="Share"/>
            </button>
          </div>
        </div>
      </div>
      <div id="sample">
        <div className="e-card">
          <div className="e-card-header-title">Eiffel Tower</div>
          <div className="e-card-content">
            The Eiffel Tower is acknowledged as the universal symbol of
            Paris and France.
          </div>
          <div className="e-card-actions e-card-vertical">
            <button className="e-card-btn">More</button>
            <a href="#">Share</a>
          </div>
        </div>
      </div>
    </div>
  );
}

```

```

    </div>);
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
function ReactApp () {
  return (
    <div>
      <div id="sample">
        <div className="e-card">
          <div className="e-card-header-title">Eiffel Tower</div>
          <div className="e-card-content">
            The Eiffel Tower is acknowledged as the universal symbol of
            Paris and France.
          </div>
          <div className="e-card-actions">
            <button className="e-card-btn">
              <img src='./fav.png' title="Bookmark" />
            </button>
            <button className="e-card-btn">
              <img src='./like.png' title="Like" />
            </button>
            <button className="e-card-btn">
              <img src='./share.png' title="Share" />
            </button>
          </div>
        </div>
      </div>
      <div id="sample">
        <div className="e-card">
          <div className="e-card-header-title">Eiffel Tower</div>
          <div className="e-card-content">
            The Eiffel Tower is acknowledged as the universal symbol of
            Paris and France.
          </div>
          <div className="e-card-actions e-card-vertical">
            <button className="e-card-btn">More</button>
            <a href="#">Share</a>
          </div>
        </div>
      </div>
    </div>
  );
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
  <title>Syncfusion React Card Sample</title>
  <meta charset="utf-8" />

```

```

<meta name="viewport" content="width=device-width, initial-scale=1.0" />
<meta name="description" content="Essential JS 2 for React Components"
/>
<meta name="author" content="Syncfusion" />
<link href="https://cdn.syncfusion.com/ej2/20.3.56/ej2-
layouts/styles/material.css" rel="stylesheet" />
<link href="index.css" rel="stylesheet" />
<script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
<script src="systemjs.config.js"></script>
</head>
<body>
  <div id='element'>
    <div id='loader'>Loading....</div>
  </div>
</body>
</html>

```

See Also

- [How to integrate other component inside the card](#)

Horizontal in React Card component

By default, all the card elements are aligned vertically one after the other as in the DOM.

You can achieve the element to align horizontally as well by adding the class `e-card-horizontal` in the root card element.

Stacked cards

- An horizontally aligned card can push a specific column to align vertical using `e-card-stacked` class.

This will align the stacked section vertically aligned differentiating from horizontal layout.

Class | Description

`e-card-horizontal` | To align card elements horizontally.

`e-card-stacked` | To align elements vertically within the horizontal layout.

```

<div className="e-card e-card-horizontal">
   --> Aligned in horizontal
  <div className="e-card-stacked"> --> Aligned in horizontal
  // Inside the element all are aligned vertical directions
</div>
</div>

```


[Class-component]**INDEX.JSX**

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component {
  render() {
    return (
      <div style={{ margin: `50px`, display: `flex`,
        flexDirection: `row`, justifyContent: `center` }}>
        <div className="e-card e-card-horizontal" style={{ width: `400px`
        }}>
          
          <div className="e-card-stacked">
            <div className="e-card-header">
              <div className="e-card-header-caption">
                <div className="e-card-header-title">Philips
Trimmer</div>
              </div>
            </div>
            <div className="e-card-content">
              Powered by the innovative DuraPower Technology which
              optimizes power consumption, Philips trimmers are designed to last longer
              than 4 ordinary trimmers.
            </div>
          </div>
        </div>
      </div>);
  }
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component<{}, {}> {
  public render() {
    return (
      <div style={{ margin: `50px`, display: `flex`, flexDirection: `row`,
        justifyContent: `center` }}>
        <div className="e-card e-card-horizontal" style={{ width: `400px`
        }}>
          
          <div className="e-card-stacked">
            <div className="e-card-header">
              <div className="e-card-header-caption">
                <div className="e-card-header-title">Philips
Trimmer</div>
              </div>
            </div>
            <div className="e-card-content">

```

```

        Powered by the innovative DuraPower Technology which
        optimizes power consumption, Philips trimmers are designed to last longer
        than 4 ordinary trimmers.
    </div>
</div>
</div>
</div>
);
}
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));
{% endraw %}

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
    <title>Syncfusion React Card Sample</title>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta name="description" content="Essential JS 2 for React Components"
/>
    <meta name="author" content="Syncfusion" />
    <link href="https://cdn.syncfusion.com/ej2/20.3.56/ej2-
layouts/styles/material.css" rel="stylesheet" />
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
    <script src="systemjs.config.js"></script>
</head>
<body>
    <div id='element'>
        <div id='loader'>Loading....</div>
    </div>
</body>
</html>

```

[Functional-component]

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
function ReactApp() {
    return <div style={{ margin: `50px`, display: `flex`, flexDirection:
`row`, justifyContent: `center` }}>
        <div className="e-card e-card-horizontal" style={{ width: `400px`
}}>
            
            <div className="e-card-stacked">
                <div className="e-card-header">
                    <div className="e-card-header-caption">
                        <div className="e-card-header-title">Philips
Trimmer</div>

```

```

        </div>
      </div>
      <div className="e-card-content">
        Powered by the innovative DuraPower Technology which
        optimizes power consumption, Philips trimmers are designed to last longer
        than 4 ordinary trimmers.
      </div>
    </div>
  </div>
</div>);
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
function ReactApp () {
  return (
    <div style={{ margin: `50px`, display: `flex`, flexDirection: `row`,
    justifyContent: `center` }}>
      <div className="e-card e-card-horizontal" style={{ width: `400px`
    }}>
        
        <div className="e-card-stacked">
          <div className="e-card-header">
            <div className="e-card-header-caption">
              <div className="e-card-header-title">Philips
Trimmer</div>
            </div>
          </div>
          <div className="e-card-content">
            Powered by the innovative DuraPower Technology which
            optimizes power consumption, Philips trimmers are designed to last longer
            than 4 ordinary trimmers.
          </div>
        </div>
      </div>
    </div>
  );
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));
{% endraw %}

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
  <title>SynCFusion React Card Sample</title>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <meta name="description" content="Essential JS 2 for React Components"
/>

```

```
<meta name="author" content="Syncfusion" />
<link href="https://cdn.syncfusion.com/ej2/20.3.56/ej2-
layouts/styles/material.css" rel="stylesheet" />
<script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
<script src="systemjs.config.js"></script>
</head>
<body>
  <div id='element'>
    <div id='loader'>Loading....</div>
  </div>
</body>
</html>
```

Style in React Card component

The following content provides the exact CSS structure that can be used to modify the control's appearance based on user preference.

Customizing the card

Use the following CSS to customize the card properties.

```
`css
.e-card {
background-color: aqua;
padding-left: 20px;
margin-bottom: 20px;
}
`
```

Customizing the Header element

Use the following CSS to customize the Header element properties.

```
`css
.e-card .e-card-header {
font-family: cursive;
font-style: italic;
}
`
```

Customizing the card content

Use the following CSS to customize the card content properties.

```
`css
.e-card .e-card-content {
font-size: 20px;
}
```

```
color: gray;
line-height: initial;
font-weight: normal;
}
`
```

Divider used to separate the elements inside the card

Use the following CSS to customize the Divider used to separate the elements inside the card properties.

```
`css
.e-card .e-card-separator {
padding-bottom: 30px;
}
`
```

Including image within card element

Use the following CSS to Include image within card element.

```
`css
.e-card .e-card-image {
background-image: url(images.png);
background-color: yellow;
height: 160px;
}
`
```

Including a title or caption for the image

Use the following CSS to Include a title or caption for the image.

```
`css
.e-card .e-card-image .e-card-title {
font-family: cursive;
font-style: italic;
}
`
```

To include heading image within the header

Use the following CSS to Include heading image within the header.

```
`css
.e-card .e-card-header .e-card-header-image {
height: 48px;
```

```
width: 48px;  
}
```

```
,
```

Customizing the Header main title

Use the following CSS to Customize the Header main title.

```
`css  
  
.e-card .e-card-header .e-card-header-caption .e-card-header-title {  
font-size: large;  
color: aquamarine;  
}  
,
```

Customizing the Header subtitle

Use the following CSS to Customize the Header subtitle.

```
`css  
  
.e-card .e-card-header .e-card-header-caption .e-card-sub-title {  
font-size: 20px;  
font-variant: all-petite-caps;  
}  
,
```

Including action buttons or anchor tags

Use the following CSS to Include action buttons or anchor tags.

```
`css  
  
.e-card .e-card-actions .e-card-btn {  
padding-left: 20px;  
background-color: wheat;  
}  
,
```

To align card elements horizontally

Use the following CSS to align card elements horizontally.

```
`css  
  
.e-card .e-card-horizontal {  
margin: auto;  
width: inherit;  
}
```

To align elements vertically within the horizontal layout

Use the following CSS to align elements vertically within the horizontal layout.

```
`css
.e-card .e-card-horizontal .e-card-stacked {
justify-content: flex-start;
margin: initial;
}
```

How To

Customize the card image title position in React Card component

Card Image titles are placed as always Bottom-Left Corner only, we can manually customize to placing titles anywhere over the image by adding styles.

[Class-component]

INDEX.JSX

```
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component {
  fields = { text: 'pos', value: 'id' };
  position = [
    { id: 'bottom-left', pos: 'BottomLeft' },
    { id: 'bottom-right', pos: 'BottomRight' },
    { id: 'top-left', pos: 'TopLeft' },
    { id: 'top-right', pos: 'TopRight' }
  ];
  constructor(props) {
    super(props);
    this.state = {
      isHorizontal: false,
      positionClass: 'e-card-bottom-left'
    };
  }
  onPositionChange(e) {
    this.setState({
      positionClass: 'e-card-' + e.value
    });
  }
  onDirectionChange(e) {
    const value = (e.checked) ? true : false;
    this.setState({ isHorizontal: value });
  }
  render() {
    return (
      <div>
        <br />
        <div className="row">
```

```

        <div className="col-xs-6 col-sm-6 col-lg-6 col-md-6">
            <CheckBoxComponent checked={false} label='Horizontal'
change={this.onDirectionChange = this.onDirectionChange.bind(this)} />
        </div>
        <div className="col-xs-6 col-sm-6 col-lg-6 col-md-6">
            <DropDownListComponent change={this.onPositionChange =
this.onPositionChange.bind(this)} dataSource={this.position}
fields={this.fields} placeholder="Select Position" width="300px"/>
        </div>
    </div>
    <br />
    <div id="sample row">
        <div className={'e-card ' + `${(this.state.isHorizontal) ? 'e-
card-horizontal' : ''}`}>
            <div className="e-card-image">
                <div className={'e-card-title ' +
`${this.state.positionClass}`}>Node.Js </div>
            </div>
            <div className="e-card-content">
                Node.js is a wildly popular platform for writing web
applications that has revolutionized web development in many ways, enjoying
support across the open source community as well as industry.
            </div>
        </div>
    </div>
</div>);
    }
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.TSX

```

import { ChangeEventArgs as EventArgs, CheckBoxComponent } from
'@syncfusion/ej2-react-buttons';
import { ChangeEventArgs, DropDownListComponent } from '@syncfusion/ej2-
react-dropdowns';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component<{}, { positionClass:
string, isHorizontal: boolean }> {
    public fields: object = { text: 'pos', value: 'id' };
    public position: any = [
        { id: 'bottom-left', pos: 'BottomLeft' },
        { id: 'bottom-right', pos: 'BottomRight' },
        { id: 'top-left', pos: 'TopLeft' },
        { id: 'top-right', pos: 'TopRight' }
    ];
    constructor(props: {}) {
        super(props);
        this.state = {
            isHorizontal: false,
            positionClass: 'e-card-bottom-left'
        }
    }
    public onPositionChange(e: ChangeEventArgs): void {
        this.setState({

```



```

        positionClass: 'e-card-' + e.value
    });
}
public onDirectionChange(e: EventArgs): void {
    const value: boolean = (e.checked) ? true : false;
    this.setState({ isHorizontal: value });
}
public render() {
    return (
        <div>
            <br />
            <div className="row">
                <div className="col-xs-6 col-sm-6 col-lg-6 col-md-6">
                    <CheckBoxComponent checked={false} label='Horizontal'
change={this.onDirectionChange = this.onDirectionChange.bind(this)} />
                </div>
                <div className="col-xs-6 col-sm-6 col-lg-6 col-md-6">
                    <DropDownListComponent change={this.onPositionChange =
this.onPositionChange.bind(this)} dataSource={this.position}
fields={this.fields} placeholder="Select Position" width="300px" />
                </div>
            </div>
            <br />
            <div id="sample row">
                <div className={'e-card ' + `${(this.state.isHorizontal) ? 'e-
card-horizontal' : ''}`}>
                    <div className="e-card-image">
                        <div className={'e-card-title ' +
`${this.state.positionClass}`}>Node.Js </div>
                    </div>
                    <div className="e-card-content">
                        Node.js is a wildly popular platform for writing web
applications that has revolutionized web development in many ways, enjoying
support across the open source community as well as industry.
                    </div>
                </div>
            </div>
        </div>
    );
}
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

[Functional-component]

INDEX.JSX

```

import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from "react";
import { useState } from 'react';
import * as ReactDOM from "react-dom";
function ReactApp() {
    let fields = { text: 'pos', value: 'id' };
    let position = [
        { id: 'bottom-left', pos: 'BottomLeft' },
    ]

```

```

    { id: 'bottom-right', pos: 'BottomRight' },
    { id: 'top-left', pos: 'TopLeft' },
    { id: 'top-right', pos: 'TopRight' }
  ];
  const [status, setStatus] = useState({ isHorizontal: false,
positionClass: 'e-card-bottom-left' });
  function onPositionChange(e) {
    setStatus({ isHorizontal: status.isHorizontal,
      positionClass: 'e-card-' + e.value
    });
  }
  function onDirectionChange(e) {
    const value = (e.checked) ? true : false;
    setStatus({ isHorizontal: value, positionClass: status.positionClass
  });
  }
  return (
    <div>
      <br />
      <div className="row">
        <div className="col-xs-6 col-sm-6 col-lg-6 col-md-6">
          <CheckBoxComponent checked={false} label='Horizontal'
change={onDirectionChange.bind(this)} />
        </div>
        <div className="col-xs-6 col-sm-6 col-lg-6 col-md-6">
          <DropDownListComponent change={onPositionChange.bind(this)}
dataSource={position} fields={fields} placeholder="Select Position"
width="300px" />
        </div>
      </div>
      <br />
      <div id="sample row">
        <div className={'e-card ' + `${(status.isHorizontal) ? 'e-card-
horizontal' : ''}`}>
          <div className="e-card-image">
            <div className={'e-card-title ' +
`${status.positionClass}`}>Node.Js </div>
          </div>
          <div className="e-card-content">
            Node.js is a wildly popular platform for writing web
applications that has revolutionized web development in many ways, enjoying
support across the open source community as well as industry.
          </div>
        </div>
      </div>
    </div>);
  }
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.TSX

```

import { ChangeEventArgs as EventArgs, CheckBoxComponent } from
'@syncfusion/ej2-react-buttons';
import { ChangeEventArgs, DropDownListComponent } from '@syncfusion/ej2-
react-dropdowns';
import * as React from "react";
import { useState } from 'react';

```

```

import * as ReactDOM from "react-dom";
function ReactApp () {
    let fields: object = { text: 'pos', value: 'id' };
    let position: any = [
        { id: 'bottom-left', pos: 'BottomLeft' },
        { id: 'bottom-right', pos: 'BottomRight' },
        { id: 'top-left', pos: 'TopLeft' },
        { id: 'top-right', pos: 'TopRight' }
    ];
    const [status, setStatus] = useState({ isHorizontal: false, positionClass:
'e-card-bottom-left' });

    function onPositionChange(e: ChangeEventArgs): void {
        setStatus({ isHorizontal: status.isHorizontal,
            positionClass: 'e-card-' + e.value
        });
    }

    function onDirectionChange(e: EventArgs): void {
        const value: boolean = (e.checked) ? true : false;
        setStatus({ isHorizontal: value, positionClass: status.positionClass
    });
    }

    return (
        <div>
            <br />
            <div className="row">
                <div className="col-xs-6 col-sm-6 col-lg-6 col-md-6">
                    <CheckBoxComponent checked={false} label='Horizontal'
change={onDirectionChange.bind(this)} />
                </div>
                <div className="col-xs-6 col-sm-6 col-lg-6 col-md-6">
                    <DropDownListComponent change={onPositionChange.bind(this)}
dataSource={position} fields={fields} placeholder="Select Position"
width="300px" />
                </div>
            </div>
            <br />
            <div id="sample row">
                <div className={'e-card ' + `${(status.isHorizontal) ? 'e-card-
horizontal' : ''}`}>
                    <div className="e-card-image">
                        <div className={'e-card-title ' +
`${status.positionClass}`}>Node.Js </div>
                    </div>
                    <div className="e-card-content">
                        Node.js is a wildly popular platform for writing web
                        applications that has revolutionized web development in many ways, enjoying
                        support across the open source community as well as industry.
                    </div>
                </div>
            </div>
        </div>
    );
}

```

```
}ReactDOM.render(<ReactApp />, document.getElementById("element"));
```

Integrate other component inside the card in React Card component

You can integrate any component inside the card element. Here ListView component is placed inside the card for showcasing the To-Do list.

[Class-component]

APP.JSX

APP.TSX

INDEX.JSX

```
import { ListViewComponent } from '@syncfusion/ej2-react-lists';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component {
  fields = { text: 'todoList' };
  // define the array of JSON
  todoList = [
    { todoList: 'Pay Bills' },
    { todoList: 'Call Chris' },
    { todoList: 'Meet Andrew' },
    { todoList: 'Visit Manager' },
    { todoList: 'Customer Meeting' },
  ];
  render() {
    return (
      <div>
        <div className="e-card" id="basic">
          <div className="e-card-title">To-Do List</div>
          <div className="e-card-separator"/>
          <div className="e-card-content">
            <ListViewComponent dataSource={this.todoList}
              fields={this.fields} showCheckBox={true}/>
          </div>
        </div>
      </div>
    );
  }
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));
```

INDEX.TSX

```
import { ListViewComponent } from '@syncfusion/ej2-react-lists';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class ReactApp extends React.Component<{}, {}> {
  public fields = { text: 'todoList' };
}
```

```
// define the array of JSON
public todoList: any = [
  { todoList: 'Pay Bills' },
  { todoList: 'Call Chris' },
  { todoList: 'Meet Andrew' },
  { todoList: 'Visit Manager' },
  { todoList: 'Customer Meeting' },
];
public render() {
  return (
    <div>
      <div className="e-card" id="basic">
        <div className="e-card-title">To-Do List</div>
        <div className="e-card-separator" />
        <div className="e-card-content">
          <ListViewComponent dataSource={this.todoList}
fields={this.fields} showCheckBox={true} />
        </div>
      </div>
    </div>
  );
}
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));
```

[Functional-component]

APP.JSX

APP.TSX

INDEX.JSX

```
import { ListViewComponent } from '@syncfusion/ej2-react-lists';
import * as React from "react";
import * as ReactDOM from "react-dom";
function ReactApp() {
  let fields = { text: 'todoList' };
  // define the array of JSON
  let todoList = [
    { todoList: 'Pay Bills' },
    { todoList: 'Call Chris' },
    { todoList: 'Meet Andrew' },
    { todoList: 'Visit Manager' },
    { todoList: 'Customer Meeting' },
  ];
  return (<div>
    <div className="e-card" id="basic">
      <div className="e-card-title">To-Do List</div>
      <div className="e-card-separator"/>
```

```

        <div className="e-card-content">
            <ListViewComponent dataSource={todoList} fields={fields}
showCheckBox={true}/>
        </div>
    </div>
</div>);
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

INDEX.TSX

```

import { ListViewComponent } from '@syncfusion/ej2-react-lists';
import * as React from "react";
import * as ReactDOM from "react-dom";
function ReactApp () {
    let fields = { text: 'todoList' };
    // define the array of JSON
    let todoList: any = [
        { todoList: 'Pay Bills' },
        { todoList: 'Call Chris' },
        { todoList: 'Meet Andrew' },
        { todoList: 'Visit Manager' },
        { todoList: 'Customer Meeting' },
    ];
    return (
        <div>
            <div className="e-card" id="basic">
                <div className="e-card-title">To-Do List</div>
                <div className="e-card-separator" />
                <div className="e-card-content">
                    <ListViewComponent dataSource={todoList} fields={fields}
showCheckBox={true} />
                </div>
            </div>
        </div>
    );
}
ReactDOM.render(<ReactApp />, document.getElementById("element"));

```

Carousel

Getting Started

This section explains how to create a simple [React CarouselLink to the Video](#), and configure its available functionalities in React.

To get started quickly with React Carousel component, you can check out this video:

Dependencies

The following list of dependencies are required to use the Carousel component in your application.

```
`javascript
```

```
|-- @syncfusion/ej2-react-navigations
```

```
|-- @syncfusion/ej2-react-base
```

```
|-- @syncfusion/ej2-navigations
|-- @syncfusion/ej2-base
|-- @syncfusion/ej2-buttons
`
```

Setup your development environment

To set-up a React application, choose any of the following ways. The best and easiest way is to use the [create-react-app](#). It sets up your development environment in JavaScript and improvise your application for production. Refer to the [installation instructions](#) of `create-react-app`.

```
`bash
npx create-react-app my-app
cd my-app
npm start
`
```

or

```
`bash
yarn create react-app my-app
cd my-app
yarn start
`
```

To set-up a React application in `TypeScript` environment, run the following command.

```
`bash
npx create-react-app my-app --template typescript
cd my-app
npm start
`
```

Besides using the [npm](#) package runner tool, also create an application from the `npm init`. To begin with the `npm init`, upgrade the `npm` version to `npm 6+`.

```
`bash
npm init react-app my-app
cd my-app
npm start
`
```

Adding Syncfusion packages

All the available Essential JS 2 packages are published in [npmjs.com](#) public registry.

To install `Carousel` component, use the following command

```
`bash
npm install @syncfusion/ej2-react-navigations --save
`
```

The above command installs [Carousel dependencies](#)

which are required to render the component in the `React` environment.

Adding Style sheet to the Application

Add `Carousel` component's styles as given below in `App.css`.

```
`css
@import "../node_modules/@syncfusion/ej2-base/styles/material.css";
@import "../node_modules/@syncfusion/ej2-buttons/styles/material.css";
@import "../node_modules/@syncfusion/ej2-navigations/styles/material.css";
`
```

Add Carousel to the project

Now, you can create `Carousel` component in the application. Add `Carousel` component in `src/App.ts` file using the following code snippet.

```
`ts
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective } from "@syncfusion/ej2-react-navigations";
import * as React from "react";
const App = () => {
  return (<div className='control-container'>
    <CarouselComponent>
      <CarouselItemsDirective>
        <CarouselItemDirective template='<figure class="img-container"><figcaption class="img-caption">Cardinal</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-container"><figcaption class="img-caption">Kingfisher</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-container"><figcaption class="img-caption">Keel-billed-
toucan</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-container"><img
src="https://ej2.syncfusion.com/products/images/carousel/kaohsiung.png" alt="yellow-warbler"
`
```



```

style="height:100%;width:100%;" /><figcaption class="img-caption">Yellow-
warbler</figcaption></figure>' />
<CarouselItemDirective template='<figure class="img-container"><figcaption class="img-caption">Bee-eater</figcaption></figure>' />
</CarouselItemsDirective>
</CarouselComponent>
</div>);
}
export default App;
`

```

Run the application

Run the application in the browser using the following command:

```
npm start
```

Note: You can also explore our [React Carousel example](#) that shows you how to configure the Carousel in React.

Populating items in React Carousel component

In the Carousel, slides can be rendered in two ways as follows,

- Populating items using carousel item
- Populating Items using data source

Populating items using carousel item

When rendering the Carousel component using items binding, you can assign templates for each item separately or assign a common template to each item. You can also customize the slide transition interval for each item separately. The following example code depicts the functionality as item property binding.

INDEX.JSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (<div className='control-container'>
    <CarouselComponent>
      <CarouselItemsDirective>
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
  </CarouselItemsDirective>
</CarouselComponent>
</div>;
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

INDEX.TSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (
    <div className='control-container'>
      <CarouselComponent>
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />

```

```

        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
    </CarouselItemsDirective>
</CarouselComponent>
</div>
);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

Populating Items using data source

When rendering the Carousel component using data binding, you can assign a common template only for all items using the [itemTemplate](#) property. You cannot set the interval for each item. The following example code depicts the functionality as data binding.

INDEX.JSX

```

import { CarouselComponent } from "@syncfusion/ej2-react-navigations";
import { useMemo } from "react";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
    const productItems = useMemo(() => [
        { ID: 1, Name: "Cardinal", imageName: 'cardinal' },
        { ID: 2, Name: "Kingfisher", imageName: 'hunei' },
        { ID: 3, Name: "Keel-billed-toucan", imageName: 'costa-rica' },
        { ID: 4, Name: "Yellow-warbler", imageName: 'kaohsiung' },
        { ID: 5, Name: "Bee-eater", imageName: 'bee-eater' }
    ], []);
    const itemTemplate = (props) => {
        return <figure className="img-container"><img
src={"https://ej2.syncfusion.com/products/images/carousel/" +
props.imageName + ".png"} alt={props.Name} style={{ height: "100%", width:
"100%" }} /><figcaption className="img-
caption">{props.Name}</figcaption></figure>;
    }
    return (
        <div className='control-container'>
            <CarouselComponent dataSource={productItems}
            itemTemplate={itemTemplate}></CarouselComponent>
        </div>
    );
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

INDEX.TSX

```
import { CarouselComponent } from "@syncfusion/ej2-react-navigations";
import { useMemo } from "react";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  const productItems: Record<string, string | number>[] = useMemo(() => [
    { ID: 1, Name: "Cardinal", imageName: 'cardinal' },
    { ID: 2, Name: "Kingfisher", imageName: 'hunei' },
    { ID: 3, Name: "Keel-billed-toucan", imageName: 'costa-rica' },
    { ID: 4, Name: "Yellow-warbler", imageName: 'kaohsiung' },
    { ID: 5, Name: "Bee-eater", imageName: 'bee-eater' }
  ], []);
  const itemTemplate = (props: any): JSX.Element => {
    return <figure className="img-container"><img
      src={"https://ej2.syncfusion.com/products/images/carousel/" +
        props.imageName + ".png"} alt={props.Name} style={{ height: "100%", width:
          "100%" }} /><figcaption className="img-
        caption">{props.Name}</figcaption></figure>;
  }
  return (
    <div className='control-container'>
      <CarouselComponent dataSource={productItems}
        itemTemplate={itemTemplate}></CarouselComponent>
    </div>
  );
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

Selection

The Carousel items will be populated from the first index of the Carousel items and can be customized using the following ways,

- Select an item using the property.
- Select an item using the method.

Select an item using the property

Using the [selectedIndex](#) property of the Carousel component, you can set the slide to be populated at the time of initial rendering else you can switch to the particular slide item.

INDEX.JSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (<div className='control-container'>
    <CarouselComponent selectedIndex={3}>
      <CarouselItemsDirective>
        <CarouselItemDirective template='<figure class="img-
        container"><figcaption class="img-
          caption">Cardinal</figcaption></figure>' />
      </CarouselItemsDirective>
    </CarouselComponent>
  );
}
```

```

        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
    </CarouselItemsDirective>
</CarouselComponent>
</div>);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

INDEX.TSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
    return (
        <div className='control-container'>
            <CarouselComponent selectedIndex={3}>
                <CarouselItemsDirective>
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
    </CarouselItemsDirective>
</CarouselComponent>
</div>
);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

Select an item using the method

Using the [prev](#) or [next](#) public method of the Carousel component, you can switch the current populating slide to a previous or next slide.

INDEX.JSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import { ButtonComponent } from "@syncfusion/ej2-react-buttons";
import { useRef } from "react";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
    const carouselRef = useRef<CarouselComponent>(null);
    const prevBtnClick = () => {
        carouselRef.current.prev();
    }
    const nextBtnClick = () => {
        carouselRef.current.next();
    }
    return (
        <div>
            <ButtonComponent className="e-btn" cssClass="e-info"
onClick={prevBtnClick}>Previous</ButtonComponent>
            <ButtonComponent className="e-btn" cssClass="e-info"
onClick={nextBtnClick}>Next</ButtonComponent>
            <div className='control-container'>
                <CarouselComponent ref={carouselRef}>
                    <CarouselItemsDirective>
                        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
                        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
                        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
    </CarouselItemsDirective>
  </CarouselComponent>
</div>
</div>
);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

INDEX.TSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import { ButtonComponent } from "@syncfusion/ej2-react-buttons";
import { useRef } from "react";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  const carouselRef = useRef<CarouselComponent>(null);
  const prevBtnClick = (): void => {
    carouselRef.current.prev();
  }
  const nextBtnClick = (): void => {
    carouselRef.current.next();
  }
  return (
    <div>
      <ButtonComponent className="e-btn" cssClass="e-info"
onClick={prevBtnClick}>Previous</ButtonComponent>
      <ButtonComponent className="e-btn" cssClass="e-info"
onClick={nextBtnClick}>Next</ButtonComponent>
      <div className='control-container'>
        <CarouselComponent ref={carouselRef}>
          <CarouselItemsDirective>
            <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
            <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
    </CarouselItemsDirective>
  </CarouselComponent>
</div>
</div>
);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

Partial visible slides

The Carousel component supports to show one complete slide and a partial view of adjacent (previous and next) slides at the same time. You can enable or disable the partial slides using the [partialVisible](#) property.

INDEX.JSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (<div className='control-container'>
    <CarouselComponent partialVisible={true}>
      <CarouselItemsDirective>
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />

```



```

        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
    </CarouselItemsDirective>
</CarouselComponent>
</div>);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

INDEX.TSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
    return (
        <div className='control-container'>
            <CarouselComponent partialVisible={true}>
                <CarouselItemsDirective>
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
                </CarouselItemsDirective>
            </CarouselComponent>
        </div>
    );
}

```

```

    );
  }
  const root = ReactDOM.createRoot(document.getElementById('element'));
  root.render(<App />);

```

Slide animation only applicable if the `partialVisible` is enabled.

The last slide will be displayed as a partial slide at the initial rendering when the `loop` and `partialVisible` properties are enabled.

The previous slide is not displayed at the initial rendering when the `loop` is disabled.

The following example code depicts the functionality of `partialVisible` and without `loop` functionalities.

INDEX.JSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (
    <div className='control-container'>
      <CarouselComponent partialVisible={true} loop={false}>
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
        </CarouselItemsDirective>
      </CarouselComponent>
    </div>);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

INDEX.TSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (
    <div className='control-container'>
      <CarouselComponent partialVisible={true} loop={false}>
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
        </CarouselItemsDirective>
      </CarouselComponent>
    </div>
  );
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

See Also

- [Customizing partial slides area size](#)[Link to the Video](#)

Navigators and indicators in React Carousel component

The navigators and indicators are used to transition the slides manually.

To have a glance at how to customize the React Carousel component's navigator buttons, play button, and indicators, watch this video:

Navigators

Show or hide previous and next button

In navigators, the previous and next slide transition buttons are used to perform slide transitions manually. You can show/hide the navigators using the [buttonsVisibility](#) property. The possible property values are as follows:

- **Hidden** – the navigator's buttons are not visible.
- **Visible** – the navigator's buttons are visible.
- **VisibleOnHover** – the navigator's buttons are visible only when hovering over the carousel.

The following example depicts the code to show/hide the navigators in the carousel.

INDEX.JSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  const showButtons = "Visible";
  return (
    <div className='control-container'>
      <CarouselComponent buttonsVisibility={showButtons}>
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
        </CarouselItemsDirective>
      </CarouselComponent>
    </div>);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

INDEX.TSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective,
CarouselButtonVisibility } from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  const showButtons: CarouselButtonVisibility = "Visible";
  return (
    <div className='control-container'>
      <CarouselComponent buttonsVisibility={showButtons}>
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
        </CarouselItemsDirective>
      </CarouselComponent>
    </div>
  );
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

Show previous and next button on hover

In the carousel, you can show the previous and next buttons only on mouse hover using the [buttonsVisibility](#) property. The following example depicts the code to show the navigators on mouse hover in the carousel.

INDEX.JSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
```

```

import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  const showButtons = "VisibleOnHover";
  return (<div className='control-container'>
    <CarouselComponent buttonsVisibility={showButtons}>
      <CarouselItemsDirective>
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
      </CarouselItemsDirective>
    </CarouselComponent>
  </div>);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

INDEX.TSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective,
CarouselButtonVisibility } from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  const showButtons: CarouselButtonVisibility = "VisibleOnHover";
  return (
    <div className='control-container'>
      <CarouselComponent buttonsVisibility={showButtons}>
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />

```

```

        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
    </CarouselItemsDirective>
</CarouselComponent>
</div>
);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

Previous and next button template

Template options are provided to customize the previous button using [previousButtonTemplate](#) and the next button using [nextButtonTemplate](#). The following example depicts the code for applying the template to previous and next buttons in the carousel.

INDEX.JSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import { ButtonComponent } from "@syncfusion/ej2-react-buttons";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
    const previousButtonTemplate = (props) => {
        return (<ButtonComponent className="e-btn" cssClass="e-flat e-round"
iconCss="e-icons e-chevron-left-double" />);
    }
    const nextButtonTemplate = (props) => {
        return (<ButtonComponent className="e-btn" cssClass="e-flat e-round"
iconCss="e-icons e-chevron-right-double" />);
    }
    return (<div className='control-container'>
        <CarouselComponent previousButtonTemplate={previousButtonTemplate}
nextButtonTemplate={nextButtonTemplate}>
            <CarouselItemsDirective>
                <CarouselItemDirective template='<figure class="img-container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
  </CarouselItemsDirective>
</CarouselComponent>
</div>;
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

INDEX.TSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import { ButtonComponent } from "@syncfusion/ej2-react-buttons";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  const previousButtonTemplate = (props: any): JSX.Element => {
    return (<ButtonComponent className="e-btn" cssClass="e-flat e-round"
iconCss="e-icons e-chevron-left-double" />);
  }
  const nextButtonTemplate = (props: any): JSX.Element => {
    return (<ButtonComponent className="e-btn" cssClass="e-flat e-round"
iconCss="e-icons e-chevron-right-double" />);
  }
  return (
    <div className='control-container'>
      <CarouselComponent previousButtonTemplate={previousButtonTemplate}
nextButtonTemplate={nextButtonTemplate}>
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />

```



```

        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
    </CarouselItemsDirective>
</CarouselComponent>
</div>
);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

Indicators

Show or hide indicators

In indicators, the total slides and current slide state have been depicted. You can show/hide the indicators using the [showIndicators](#) property. The following example depicts the code to show/hide the indicators in the carousel.

INDEX.JSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
    return (<div className='control-container'>
        <CarouselComponent showIndicators={true}>
            <CarouselItemsDirective>
                <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
            </CarouselItemsDirective>
        </CarouselComponent>
    </div>);
}

```

```

        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
    </CarouselItemsDirective>
</CarouselComponent>
</div>;
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

INDEX.TSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
    return (
        <div className='control-container'>
            <CarouselComponent showIndicators={true}>
                <CarouselItemsDirective>
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
                </CarouselItemsDirective>
            </CarouselComponent>
        </div>
    );
}

```

```
);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

Indicators Template

Template option is provided to customize the indicators by using the [indicatorTemplate](#) property. The following example depicts the code for applying a template to indicators in the carousel.

INDEX.JSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import { useMemo } from "react";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  const indicatorTemplate = useMemo(() => {
    return (props) => <div className="indicator" indicator-
index={props.index}></div>;
  }, []);
  return (
    <div className='control-container'>
      <CarouselComponent indicatorsTemplate={indicatorTemplate}>
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
        </CarouselItemsDirective>
      </CarouselComponent>
    </div>
  );
}
const root = ReactDOM.createRoot(document.getElementById('element'));
```

```
root.render(<App />);
```

INDEX.TSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import { useMemo } from "react";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
    const indicatorTemplate = useMemo(() => {
        return (props: any) => <div className="indicator" indicator-
index={props.index}></div>;
    }, []);
    return (
        <div className='control-container'>
            <CarouselComponent indicatorsTemplate={indicatorTemplate}>
                <CarouselItemsDirective>
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
                </CarouselItemsDirective>
            </CarouselComponent>
        </div>
    );
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

INDEX.CSS

```
.control-container {
    height: 360px;
    margin: 0 auto;
```

```

width: 600px;
}
.img-container {
  height: 100%;
  margin: 0;
}
.img-caption {
  color: #fff;
  font-size: 1rem;
  position: absolute;
  bottom: 3rem;
  width: 100%;
  text-align: center;
}
.e-carousel .e-carousel-indicators .e-indicator-bars .e-indicator-bar.e-
template .indicator {
  background-color: #ecec;
  border-radius: 0.25rem;
  cursor: pointer;
  height: 0.5rem;
  margin: 0.5rem;
  width: 1.5rem;
}
.e-carousel .e-carousel-indicators .e-indicator-bars .e-indicator-bar.e-
active .indicator {
  background-color: #3c78ef;
}

```

Showing preview of slide in indicator

You can customize the indicators by showing the preview image of each slide using the [indicatorTemplate](#) property. The following example depicts the code for showing the preview image using a template for indicators in the carousel.

INDEX.JSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import { useState } from "react";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  const [slides] = useState(["Slide 1", "Slide 2", "Slide 3", "Slide 4",
"Slide 5"]);
  const getContent = (index) => {
    return slides[index];
  }
  const indicatorTemplate = (props) => {
    return (
      <div className="indicator" indicator-index={props.index}>
        <div className="preview-content">{getContent(props.index)}</div>
      </div>
    );
  }
  return (
    <div className="control-container">
      <CarouselComponent indicatorsTemplate={indicatorTemplate}>

```

```

        <CarouselItemsDirective>
          <CarouselItemDirective template="<div class='slide-content'>Slide
1</div>" />
          <CarouselItemDirective template="<div class='slide-content'>Slide
2</div>" />
          <CarouselItemDirective template="<div class='slide-content'>Slide
3</div>" />
          <CarouselItemDirective template="<div class='slide-content'>Slide
4</div>" />
          <CarouselItemDirective template="<div class='slide-content'>Slide
5</div>" />
        </CarouselItemsDirective>
      </CarouselComponent>
    </div>
  );
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

INDEX.TSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import { useState } from "react";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  const [slides] = useState(["Slide 1", "Slide 2", "Slide 3", "Slide 4",
"Slide 5"]);
  const getContent = (index: number) => {
    return slides[index];
  }
  const indicatorTemplate = (props: any): JSX.Element => {
    return (
      <div className="indicator" indicator-index={props.index}>
        <div className="preview-content">{getContent(props.index)}</div>
      </div>
    );
  }
  return (
    <div className="control-container">
      <CarouselComponent indicatorsTemplate={indicatorTemplate}>
        <CarouselItemsDirective>
          <CarouselItemDirective template="<div class='slide-content'>Slide
1</div>" />
          <CarouselItemDirective template="<div class='slide-content'>Slide
2</div>" />
          <CarouselItemDirective template="<div class='slide-content'>Slide
3</div>" />
          <CarouselItemDirective template="<div class='slide-content'>Slide
4</div>" />
          <CarouselItemDirective template="<div class='slide-content'>Slide
5</div>" />
        </CarouselItemsDirective>
      </CarouselComponent>
    </div>
  );
}

```

```
);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

INDEX.CSS

```
.control-container {
  background-color: #adb5bd;
  height: 300px !important;
  margin: 0 auto;
  width: 500px !important;
}
.e-carousel .slide-content {
  align-items: center;
  display: flex;
  font-size: 1.25rem;
  height: 100%;
  justify-content: center;
}
.e-carousel .e-carousel-items,
.e-carousel .e-carousel-navigators {
  height: calc(100% - 3rem);
}
.e-carousel .e-carousel-navigators .e-previous,
.e-carousel .e-carousel-navigators .e-next,
.e-carousel .e-carousel-navigators .nav-btn {
  padding: 0;
}
.e-carousel .e-carousel-navigators .nav-btn:active,
.e-carousel .e-carousel-navigators .nav-btn:focus,
.e-carousel .e-carousel-navigators .nav-btn:hover {
  background-color: transparent !important;
  color: inherit;
}
.e-carousel .e-carousel-navigators svg {
  fill: none;
  stroke: currentColor;
  stroke-linecap: square;
  stroke-width: 8px;
  height: 2rem;
  vertical-align: middle;
  width: 2rem;
}
.e-carousel .e-carousel-navigators .e-previous svg {
  transform: rotate(180deg);
}
.e-carousel .e-carousel-indicators .e-indicator-bars .e-indicator-bar
.indicator {
  background-color: #adb5bd;
  border: 1px solid black;
  border-radius: 0.25rem;
  cursor: pointer;
  height: 3.5rem;
  margin: 0.5rem;
  width: 5rem;
```

```

}
.e-carousel .e-carousel-indicators .e-indicator-bars .e-indicator-bar.e-
active .indicator {
  background-color: #c1cdda;
}
.preview-content {
  align-items: center;
  display: flex;
  height: 100%;
  justify-content: center;
}

```

Indicators Types

Choose different types of indicators available using the [indicatorsType](#) property. The indicator types are categorized as follows:

- [Default Indicator](#)
- [Dynamic Indicator](#)
- [Fraction Indicator](#)
- [Progress Indicator](#)

Default Indicator

A default indicator in a carousel is a set of dots that indicate the current position of the slide in the carousel. The Default indicator can be achieved by setting the [indicatorsType](#) to **Default**.

INDEX.JSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (
    <div className='control-container'>
      <CarouselComponent indicatorsType="Default">
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><img
src="https://ej2.syncfusion.com/products/images/carousel/kaohsiung.png"

```



```

alt="yellow-warbler" style="height:100%;width:100%;" /><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
    </CarouselItemsDirective>
</CarouselComponent>
</div>);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

INDEX.TSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
    return (
        <div className='control-container'>
            <CarouselComponent indicatorsType="Default">
                <CarouselItemsDirective>
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
                </CarouselItemsDirective>
            </CarouselComponent>
        </div>
    );
}
const root = ReactDOM.createRoot(document.getElementById('element'));

```

```
root.render(<App />);
```

Dynamic Indicator

A dynamic indicator in a carousel provides visual cues or markers that dynamically change or update to indicate the current position. The Dynamic indicator can be achieved by setting the [indicatorsType](#) to **Dynamic**.

INDEX.JSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
  from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (<div className='control-container'>
    <CarouselComponent indicatorsType="Dynamic">
      <CarouselItemsDirective>
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>'>
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>'>
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>'>
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>'>
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>'>
      </CarouselItemsDirective>
    </CarouselComponent>
  </div>);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

INDEX.TSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
  from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
```

```

const App = () => {
  return (
    <div className='control-container'>
      <CarouselComponent indicatorsType="Dynamic">
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
        </CarouselItemsDirective>
      </CarouselComponent>
    </div>
  );
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

Fraction Indicator

The fraction indicator type displays the current slide index and total slide count as a fraction. The Fraction indicator can be achieved by setting the [indicatorsType](#) to **Fraction**.

INDEX.JSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (<div className='control-container'>
    <CarouselComponent indicatorsType="Fraction">
      <CarouselItemsDirective>
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
  </CarouselItemsDirective>
</CarouselComponent>
</div>;
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

INDEX.TSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (
    <div className='control-container'>
      <CarouselComponent indicatorsType="Fraction">
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />

```

```

        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
    </CarouselItemsDirective>
</CarouselComponent>
</div>
);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

Progress Indicator

The Progress Indicator type displays the current slide as a progress bar. The Progress indicator can be achieved by setting the [indicatorsType](#) to **Progress**.

INDEX.JSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
    return (<div className='control-container'>
        <CarouselComponent indicatorsType="Progress">
            <CarouselItemsDirective>
                <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
    </CarouselItemsDirective>
  </CarouselComponent>
</div>);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

INDEX.TSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (
    <div className='control-container'>
      <CarouselComponent indicatorsType="Progress">
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
        </CarouselItemsDirective>
      </CarouselComponent>
    </div>
  );
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

Play button

Show or hide the play button

In the carousel, [autoPlay](#) actions have been controlled by using the [showPlayButton](#) property in the user interface. When you enable this property, the slide transitions are controlled using this play and pause button. This property depends on the [buttonsVisibility](#) property. The following example depicts the code to show the play button in the carousel.

INDEX.JSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
    return (<div className='control-container'>
        <CarouselComponent showPlayButton={true}>
            <CarouselItemsDirective>
                <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
            </CarouselItemsDirective>
        </CarouselComponent>
    </div>);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

INDEX.TSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
```

```

const App = () => {
  return (
    <div className='control-container'>
      <CarouselComponent showPlayButton={true}>
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
        </CarouselItemsDirective>
      </CarouselComponent>
    </div>
  );
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

Play button template

Template option is provided to customize the play button by using the [playButtonTemplate](#) property. The following example depicts the code for applying a template to play Button in the carousel.

INDEX.JSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import { ButtonComponent } from "@syncfusion/ej2-react-buttons";
import * as React from "react";
import { useState } from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  const [buttonContent, setButtonContent] = useState('Pause');
  const [autoPlay, setAutoPlay] = useState(true);
  const btnClick = () => {
    if (autoPlay) {
      setButtonContent('Play');
    }
  }
}

```



```

        setAutoPlay(false);
    } else {
        setButtonContent('Pause');
        setAutoPlay(true);
    }
}

const playButtonTemplate = (props) => {
    return (
        <ButtonComponent className="e-btn" cssClass="e-info playBtn"
content={buttonContent} onClick={btnClick} />
    );
}
return (
    <div className='control-container'>
        <CarouselComponent showPlayButton={true}
playButtonTemplate={playButtonTemplate} autoPlay={autoPlay}>
            <CarouselItemsDirective>
                <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
            </CarouselItemsDirective>
        </CarouselComponent>
    </div>
    );
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

INDEX.TSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import { ButtonComponent } from "@syncfusion/ej2-react-buttons";

```

```

import * as React from "react";
import { useState } from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  const [buttonContent, setButtonContent] = useState<string>('Pause');
  const [autoPlay, setAutoPlay] = useState<boolean>(true);
  const btnClick = () => {
    if (autoPlay) {
      setButtonContent('Play');
      setAutoPlay(false);
    } else {
      setButtonContent('Pause');
      setAutoPlay(true);
    }
  }

  const playButtonTemplate = (props: any): JSX.Element => {
    return (
      <ButtonComponent className="e-btn" cssClass="e-info playBtn"
content={buttonContent} onClick={btnClick} />
    );
  }

  return (
    <div className='control-container'>
      <CarouselComponent showPlayButton={true}
playButtonTemplate={playButtonTemplate} autoPlay={autoPlay}>
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
        </CarouselItemsDirective>
      </CarouselComponent>
    </div>
  );
}

```

```
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

Animations and transitions in React Carousel component

Animations

Fade animation

In Carousel, two built-in animations are provided for slide transitions. You can disable animation using the [animationEffect](#) property. By default, Slide animation is applied for the transition between slides.

The following demo depicts the example for fade animation,

INDEX.JSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
    return (<div className='control-container'>
        <CarouselComponent animationEffect="Fade">
            <CarouselItemsDirective>
                <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
            </CarouselItemsDirective>
        </CarouselComponent>
    </div>);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

INDEX.TSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (
    <div className='control-container'>
      <CarouselComponent animationEffect="Fade">
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
        </CarouselItemsDirective>
      </CarouselComponent>
    </div>
  );
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

Custom animation

In Carousel, you can use customized animation effects for slide transitions using the [Custom](#) option of the [animationEffect](#) property and apply custom animation css via [cssClass](#) property.

The following demo depicts the example for **parallax** custom animation,

INDEX.JSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (<div className='control-container'>
```

```

        <CarouselComponent animationEffect="Custom" cssClass="parallax">
          <CarouselItemsDirective>
            <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
            <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
            <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
            <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
            <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
          </CarouselItemsDirective>
        </CarouselComponent>
      </div>;
    }
    const root = ReactDOM.createRoot(document.getElementById('element'));
    root.render(<App />);

```

INDEX.TSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (
    <div className='control-container'>
      <CarouselComponent animationEffect= "Custom" cssClass= "parallax">
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />

```

```

        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
    </CarouselItemsDirective>
  </CarouselComponent>
</div>
);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

Intervals between slides

Using the items property, you can set different intervals for each item to transition between slides. The default interval is 5000 ms (5 seconds). The following example depicts the code for setting the different intervals between each item.

INDEX.JSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective, }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (<div className='control-container'>
    <CarouselComponent>
      <CarouselItemsDirective>
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' interval={3000}/>
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' interval={1000}/>
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>'
interval={2000}/>
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' interval={5000}/>
    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' interval={6000}/>
    </CarouselItemsDirective>
  </CarouselComponent>
</div>;
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

INDEX.TSX

```

import {
  CarouselComponent,
  CarouselItemsDirective,
  CarouselItemDirective,
} from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (
    <div className='control-container'>
      <CarouselComponent>
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' interval={3000} />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' interval={1000} />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>'
interval={2000} />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' interval={5000} />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' interval={6000} />
        </CarouselItemsDirective>
      </CarouselComponent>
    </div>
  );
};

```

```

    </CarouselComponent>
  </div>
);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

Note: Interval property can accept value in terms of milliseconds.

Auto play slides

In the carousel, all slides transitions are performed continuously after the specified or default intervals. You can enable or disable the auto slide transition using the [autoPlay](#) property. The following example depicts the code to enable or disable the auto slide transitions.

INDEX.JSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (<div className='control-container'>
    <CarouselComponent autoPlay={true}>
      <CarouselItemsDirective>
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
      </CarouselItemsDirective>
    </CarouselComponent>
  </div>);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```


INDEX.TSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (
    <div className='control-container'>
      <CarouselComponent autoPlay={true}>
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
        </CarouselItemsDirective>
      </CarouselComponent>
    </div>
  );
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

Pause on hover

By default, Slide transitions are paused when hovering the mouse pointer over the Carousel element. You can enable or disable this functionality using the [pauseOnHover](#) property.

The following example depicts the code to play the slides when hovering the mouse pointer over the Carousel element.

INDEX.JSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
```

```
import * as ReactDOM from "react-dom";
const App = () => {
  return (<div className='control-container'>
    <CarouselComponent pauseOnHover={false}>
      <CarouselItemsDirective>
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
      </CarouselItemsDirective>
    </CarouselComponent>
  </div>);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

INDEX.TSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (
    <div className='control-container'>
      <CarouselComponent pauseOnHover={false}>
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
    </CarouselItemsDirective>
  </CarouselComponent>
</div>
);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

Looping slides

In the carousel, slides transitions are repeated continuously when you reach the last slide by default. You can enable or disable the infinite slide transition using the [loop](#) property. The following example depicts the code to enable or disable the infinite slide transitions.

INDEX.JSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective, }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (<div className='control-container'>
    <CarouselComponent loop={true}>
      <CarouselItemsDirective>
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />

```

```

        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
    </CarouselItemsDirective>
</CarouselComponent>
</div>;
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

INDEX.TSX

```

import {
    CarouselComponent,
    CarouselItemsDirective,
    CarouselItemDirective,
} from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
    return (
        <div className='control-container'>
            <CarouselComponent loop={true}>
                <CarouselItemsDirective>
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
                    <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
                </CarouselItemsDirective>
            </CarouselComponent>
        </div>
    );
}

```

```

        </CarouselItemsDirective>
      </CarouselComponent>
    </div>
  );
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

Slide changing events

Using the [slideChanging](#) or [slideChanged](#) events of the Carousel component, you can perform sample end customization while the carousel items are switched.

The following demo depicts the example for carousel events,

INDEX.JSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective }
from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
const App = () => {
  const onSlideChanging = (args) => {
    console.log(args.currentSlide); // You can customize the slide
    before changing.
  };
  const onSlideChanged = (args) => {
    console.log(args.currentSlide); // You can customize the slide after
    changed.
  };
  return (
    <div className='control-container'>
      <CarouselComponent slideChanging={onSlideChanging}
      slideChanged={onSlideChanged}>
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
          container"><figcaption class="img-
          caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
          container"><figcaption class="img-
          caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
          container"><figcaption
          class="img-caption">Keel-billed-toucan</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
          container"><figcaption
          class="img-caption">Yellow-warbler</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
          container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
    </CarouselItemsDirective>
  </CarouselComponent>
</div>);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

INDEX.TSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective,
SlideChangingEventArgs, SlideChangedEventArgs } from '@syncfusion/ej2-react-
navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
const App = () => {
  const onSlideChanging = (args: SlideChangingEventArgs): void => {
    console.log(args.currentSlide); // You can customize the slide before
    changing.
  }
  const onSlideChanged = (args: SlideChangedEventArgs): void => {
    console.log(args.currentSlide); // You can customize the slide after
    changed.
  }
  return (
    <div className='control-container'>
      <CarouselComponent slideChanging={onSlideChanging}
slideChanged={onSlideChanged}>
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
        </CarouselItemsDirective>
```

```

    </CarouselComponent>
  </div>
);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

Disable touch swiping

In the carousel, you can swipe the carousel slides using touch actions by default. The swipe action can be enabled or disabled using the [enableTouchSwipe](#) property. The following example depicts the code to disable the swipe action for the slide.

INDEX.JSX

```

import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective, }
from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (<div className='control-container'>
    <CarouselComponent enableTouchSwipe={false}>
      <CarouselItemsDirective>
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
        <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
      </CarouselItemsDirective>
    </CarouselComponent>
  </div>);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

INDEX.TSX

```

import {
  CarouselComponent,
  CarouselItemsDirective,
  CarouselItemDirective,
} from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
  return (
    <div className='control-container'>
      <CarouselComponent enableTouchSwipe={false}>
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
        </CarouselItemsDirective>
      </CarouselComponent>
    </div>
  );
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

Swipe Modes

In the carousel, the [swipeMode](#) property allows specifying whether the slide transition should occur while performing swiping via touch or mouse. The slide swiping is enabled or disabled using the bitwise operator.

The following are the different swipe modes available in the carousel:

- `CarouselSwipeMode.Touch` - Allows the user to slide the slides using touch actions.
- `CarouselSwipeMode.Mouse` - Allows the user to slide the slides using mouse actions.

- CarouselSwipeMode.Touch & CarouselSwipeMode.Mouse - Allows the user to slide the slides using both touch and mouse actions.
- ~CarouselSwipeMode.Touch & ~CarouselSwipeMode.Mouse - Disables both touch and mouse actions.

INDEX.JSX

```
import { CarouselComponent, CarouselItemsDirective, CarouselItemDirective,
CarouselSwipeMode } from "@syncfusion/ej2-react-navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
const App = () => {
    return (<div className='control-container'>
        <CarouselComponent swipeMode={CarouselSwipeMode.Touch &
CarouselSwipeMode.Mouse}>
            <CarouselItemsDirective>
                <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
                <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
            </CarouselItemsDirective>
        </CarouselComponent>
    </div>);
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);
```

INDEX.TSX

```
import { CarouselComponent, CarouselItemsDirective,
CarouselItemDirective, CarouselSwipeMode } from "@syncfusion/ej2-react-
navigations";
import * as React from "react";
import * as ReactDOM from "react-dom";
```

```

const App = () => {
  return (
    <div className='control-container'>
      <CarouselComponent swipeMode={CarouselSwipeMode.Touch &
CarouselSwipeMode.Mouse}>
        <CarouselItemsDirective>
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Cardinal</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Kingfisher</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Keel-billed-toucan</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption
class="img-caption">Yellow-warbler</figcaption></figure>' />
          <CarouselItemDirective template='<figure class="img-
container"><figcaption class="img-
caption">Bee-eater</figcaption></figure>' />
        </CarouselItemsDirective>
      </CarouselComponent>
    </div>
  );
}
const root = ReactDOM.createRoot(document.getElementById('element'));
root.render(<App />);

```

Accessibility in React Carousel component

The Carousel component has been designed, keeping in mind the [WAI-ARIA](#) specifications, and applying the WAI-ARIA roles, states and properties along with keyboard support for people who use assistive devices. WAI-ARIA accessibility support is achieved through attributes like `aria-roledescription`, `aria-label`, `aria-current`, `aria-live`, `aria-role` and `aria-hidden`. It provides information about elements in a document for assistive technology. The component implements keyboard navigation support by following the [WAI-ARIA practices](#) and has been tested in major screen readers.

The accessibility compliance for the Carousel component is outlined below.

| Accessibility Criteria | Compatibility |

| -- | -- |

| [WCAG 2.2](#) Support | |

| [Section 508](#) Support | |

| Screen Reader Support | |

| Right-To-Left Support | |

| Color Contrast | |

| Mobile Device Support | |

| Keyboard Navigation Support | |

| [Accessibility Checker](#) Validation | |

| [Axe-core](#) Accessibility Validation | |

```
<style>
.post .post-content img {
display: inline-block;
margin: 0.5em 0;
}
</style>

<div> - All
features of the component meet the requirement.</div>

<div> - Some features of the component do not meet the requirement.</div>

<div> - The component does not meet the requirement.</div>
```

ARIA attributes

The Carousel component is designed by considering [WAI-ARIA](#) standard. Carousel is supported with ARIA Accessibility which is accessible by on-screen readers and other assistive technology devices. The following list of attributes is added to the Carousel.

| Roles and Attributes | Functionalities

Roles and Attributes	Functionalities
aria-roledescription	The role description attribute has been added for the root element (Carousel) and each Carousel slide item (slide).

aria-label	Previous, next and play/pause buttons and all indicator elements.
aria-current	For the active item indicator element, <code>aria-current</code> is set to <code>true</code> .
aria-hidden	For all Carousel elements except the currently visible item, <code>aria-hidden</code> is set to <code>true</code> .
aria-live	For Carousel items element, when <code>autoPlay</code> is <code>true</code> , <code>aria-live</code> is set to <code>off</code> ; when <code>autoPlay</code> is <code>false</code> , <code>aria-live</code> is set to <code>polite</code> .
aria-role	For Carousel slide item, <code>aria-role</code> has been grouped.

Keyboard interaction

By default, keyboard navigation is enabled. This component implements keyboard navigation support by following the WAI-ARIA practices. Once focused on the active Carousel element, you can use the following key combination for interacting with the Carousel.

Key	Description	
-----	-----	
Alt + J	Keys to focus the Carousel component (done at application end).	
Arrows	Keys to navigate between slides.	
Home	To navigate to the first slide.	
End	To navigate to the last slide.	
Space	To play/pause the slide transitions.	
Enter	To perform the respective action on its focus.	
Tab	To Move focus through the interactive elements.	
Shift + Tab	To Move focus through the interactive elements.	

Ensuring accessibility

The Carousel component's accessibility levels are ensured through an [accessibility-checker](#) and [axe-core](#) software tools during automated testing.

The accessibility compliance of the Carousel component is shown in the following sample. Open the [sample](#) in a new window to evaluate the accessibility of the Carousel component with accessibility tools.

See also

- [Accessibility in Syncfusion React components](#)

Styles and appearance in React Carousel component

To modify the Carousel appearance, you need to override the default CSS of Carousel component.

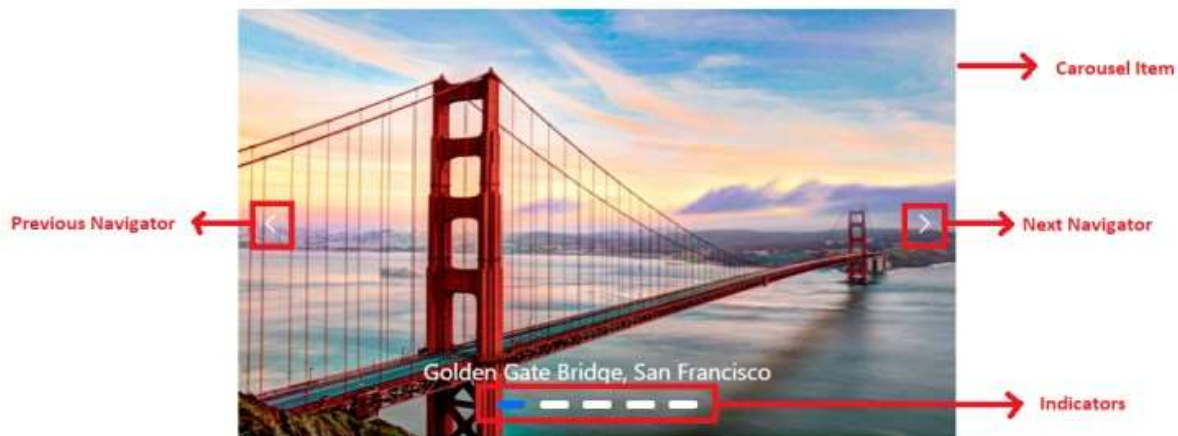
Please find the list of CSS classes and its corresponding section in Carousel component. Also, you have an option to create your own custom theme for the controls using our [Theme Studio](#).

CSS Structure in React Carousel Control

The following content provides the exact CSS structure that can be used to modify the control's appearance based on user preference.

CSS Class | Purpose of Class

- |.e-carousel|.e-carousel-item|To customize the carousel item
- |.e-carousel-item.e-active|To customize the active carousel item
- |.e-carousel|.e-carousel-indicators|To customize the indicators
- |.e-carousel|.e-carousel-indicators|.e-indicator-bars|.e-indicator-bar|To customize the indicator bars
- |.e-carousel|.e-carousel-indicators|.e-indicator-bars|.e-indicator-bar|.e-indicator|To customize the individual indicator appearance
- |.e-carousel|.e-carousel-navigators|To customize the navigators
- |.e-carousel|.e-carousel-navigators|.e-previous|To customize the previous button
- |.e-carousel|.e-carousel-navigators|.e-next|To customize the next button
- |.e-carousel|.e-carousel-navigators|.e-play-pause|To customize the play and pause button
- |.e-carousel.e-partial|.e-carousel-slide-container|To customize the partial visible slides



Customizing the indicators

Use the following CSS to customize the space between indicators by overriding the `.e-indicator-bar` CSS class.

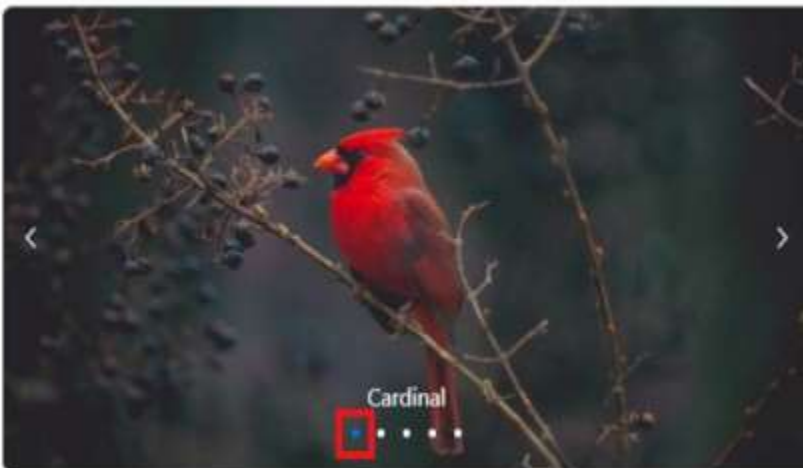
```
`css
```

```
.e-carousel .e-carousel-indicators .e-indicator-bars .e-indicator-bar {  
padding: 8px;  
}  
`
```



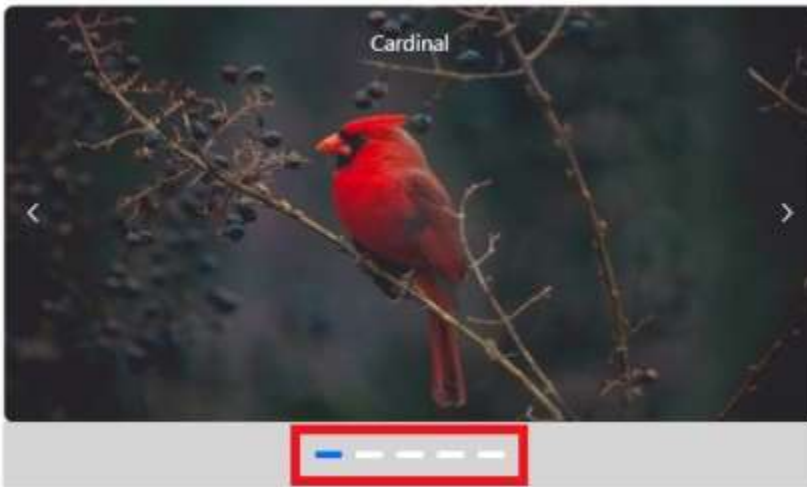
Use the following CSS to customize the indicators appearance by overriding the `.e-indicator` CSS class.

```
`css
.e-carousel .e-carousel-indicators .e-indicator-bars .e-indicator-bar .e-indicator {
width: 20px;
border-radius: 100%;
}
`
```



Use the following CSS to render the indicators outside the carousel items by overriding the `.e-carousel-indicators` CSS class.

```
`css
.e-carousel .e-carousel-indicators {
bottom: auto;
}
`
```



Customizing the navigators

Use the following CSS to customize the previous and next icon size and colors.

```
`css
.e-carousel .e-carousel-navigators .e-next .e-btn:not(:disabled) .e-btn-icon,
.e-carousel .e-carousel-navigators .e-previous .e-btn:not(:disabled) .e-btn-icon
{
  color: greenyellow;
  font-size: 25px;
}
```



Use the following CSS to customize the navigators position to bottom by overriding the `.e-carousel-navigators` CSS class.

```
`css
```

```
.e-carousel .e-carousel-navigators {  
top: 120px;  
}  
、
```



Use the following CSS to render the previous and next icon to outside the carousel items by overriding the `.e-previous` and `.e-next` CSS class.

```
`css  
.e-carousel .e-carousel-navigators .e-previous,  
.e-carousel .e-carousel-navigators .e-next  
{  
margin: -60px;  
background: black;  
}  
、
```




Customizing partial slides size

You can customize the partial slide size by overriding the `.e-carousel-slide-container` CSS class.

```
`css
.e-carousel.e-partial .e-carousel-slide-container{
padding: 0 150px;
}
```



[Link to the Video](#)

Chart

Getting Started

<!-- markdownlint-disable MD036 -->

Getting Started

This section explains you the steps required to create a simple chart and demonstrate the basic usage of the chart control.

To get start quickly with React Charts, you can check on this video:

[Dependencies](#)

Below is the list of minimum dependencies required to use the chart component.

```
`javascript
|-- @syncfusion/ej2-react-charts
|-- @syncfusion/ej2-base
|-- @syncfusion/ej2-data
|-- @syncfusion/ej2-charts
|-- @syncfusion/ej2-react-base
|-- @syncfusion/ej2-pdf-export
|-- @syncfusion/ej2-file-utils
|-- @syncfusion/ej2-compression
|-- @syncfusion/ej2-svg-base
`,`
```

[Installation and configuration](#)

You can use [create-react-app](#) to setup the applications.

To install `create-react-app` run the following command.

```
`
npm install -g create-react-app
`,`
```

- To set-up a React application in TypeScript environment, run the following command.

```
`
create-react-app quickstart --template typescript
```

```
cd quickstart
```

```
npm start
`,`
```

- To set-up a React application in JavaScript environment, run the following command.

```
`
create-react-app quickstart
```

```
cd quickstart
```

```
npm start
`,`
```

- Install Syncfusion packages using below command.

```
npm install @syncfusion/ej2-react-charts --save
```

Add Chart to the Project

Now, you can start adding Chart component in the application.

For getting started, add the Chart component in `src/App.tsx` file using following code.

INDEX.JSX

```
import { ChartComponent } from '@syncfusion/ej2-react-charts';
import * as React from 'react';
function App() {
  return (<ChartComponent />);
}
export default App;
```

INDEX.TSX

```
import { ChartComponent } from '@syncfusion/ej2-react-charts';
import * as React from 'react';
function App() {
  return (<ChartComponent />);
}
export default App;
```

Now run the `npm start` command in the console, it will run your application and open the browser window.

```
npm start
```

The below example shows a basic Chart.

INDEX.JSX

```
import { ChartComponent } from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  return <ChartComponent id='charts' />;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import { ChartComponent } from '@syncfusion/ej2-react-charts';
```

```
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  return <ChartComponent id='charts' />
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Module Injection

Chart component are segregated into individual feature-wise modules. In order to use a particular feature, you need to inject its feature service in the AppModule. In the current application, we are going to modify the above basic chart to visualize sales data for a particular year. For this application we are going to use line series, tooltip, data label, category axis and legend feature of the chart. Please find the relevant feature service name and description as follows.

- **LineSeries** - Inject this module in to **services** to use line series.
- **Legend** - Inject this module in to **services** to use legend feature.
- **Tooltip** - Inject this module in to **services** to use tooltip feature.
- **DataLabel** - Inject this module in to **services** to use datalabel feature.
- **Category** - Inject this module in to **services** to use category feature.

These modules should be injected to the **services** section as follows,

INDEX.JSX

```
import { Category, ChartComponent, DataLabel, LineSeries, Legend, Tooltip,
Inject } from '@syncfusion/ej2-react-charts';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  return <ChartComponent id='charts'>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category]} />
  </ChartComponent>;
}
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import { Category, ChartComponent, DataLabel, LineSeries, Legend, Tooltip,
Inject } from '@syncfusion/ej2-react-charts';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  return <ChartComponent id='charts'>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category]} />
  </ChartComponent>;
}
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Populate Chart with Data

This section explains how to plot below JSON data to the chart.

INDEX.JSX

```
export let data = [
  { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
  { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
  { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
  { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
  { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
  { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
];
```

INDEX.TSX

```
export let data: any[] = [
  { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
  { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
  { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
  { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
  { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
  { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
];
```

Add a series object to the chart by using [series](#) property. Now map the field names `month` and `sales` in the JSON data to the [xName](#) and [yName](#) properties of the series, then set the JSON data to [dataSource](#) property.

Since the JSON contains category data, set the [valueType](#) for horizontal axis to `Category`. By default, the axis `valueType` is `Numeric`.

INDEX.JSX

```
import { Category, ChartComponent, ColumnSeries, Inject, LineSeries,
SeriesCollectionDirective, SeriesDirective, Tooltip } from '@syncfusion/ej2-react-charts';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  const data = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const primaryXAxis = { valueType: 'Category' };
  return <ChartComponent id="charts" primaryXAxis={primaryXAxis}>
    <Inject services={[ColumnSeries, Tooltip, LineSeries, Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='month' yName='sales'
name='Sales' />
    </SeriesCollectionDirective>
  </ChartComponent>;
```

```

}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import { AxisModel, Category, ChartComponent, ColumnSeries, Inject,
LineSeries, SeriesCollectionDirective, SeriesDirective, Tooltip } from
'@syncfusion/ej2-react-charts';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  const data: any[] = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category' };
  return <ChartComponent id="charts" primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, Tooltip, LineSeries, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='month' yName='sales'
name='Sales' />
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

The sales data are in thousands, so format the vertical axis label by adding \$ as a prefix and K as a suffix to each label. This can be achieved by setting the ``${value}K`` to the `labelFormat` property of axis. Here, `{value}` act as a placeholder for each axis label.

INDEX.JSX

```

import { Category, ChartComponent, ColumnSeries, Inject, LineSeries,
SeriesCollectionDirective, SeriesDirective, Tooltip } from '@syncfusion/ej2-
react-charts';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  const data = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const primaryxAxis = { valueType: 'Category' };
  const primaryyAxis = { labelFormat: `${value}K` };

```

```

    return <ChartComponent id="charts" primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}>
      <Inject services={[ColumnSeries, Tooltip, LineSeries, Category]}/>
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='month' yName='sales'
        name='Sales' />
      </SeriesCollectionDirective>
    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import { AxisModel, Category, ChartComponent, ColumnSeries, Inject,
LineSeries, SeriesCollectionDirective, SeriesDirective, Tooltip } from
'@syncfusion/ej2-react-charts';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  const data: any[] = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category' };
  const primaryyAxis: AxisModel = { labelFormat: '${value}K' };
  return <ChartComponent id="charts" primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}>
    <Inject services={[ColumnSeries, Tooltip, LineSeries, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='month' yName='sales'
      name='Sales' />
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Add Chart Title

You can add a title using [title](#) property to the chart to provide quick information to the user about the data plotted in the chart.

INDEX.JSX

```

import { Category, ChartComponent, ColumnSeries, Inject, LineSeries,
SeriesCollectionDirective, SeriesDirective, Tooltip } from '@syncfusion/ej2-
react-charts';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {

```

```

const data = [
  { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
  { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
  { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
  { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
  { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
  { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
];
const primaryxAxis = { valueType: 'Category' };
return <ChartComponent id="charts" primaryXAxis={primaryxAxis}
title='Sales Analysis'>
  <Inject services={[ColumnSeries, Tooltip, LineSeries, Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='month' yName='sales'
name='Sales' />
  </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import { AxisModel, Category, ChartComponent, ColumnSeries, Inject,
LineSeries, SeriesCollectionDirective, SeriesDirective, Tooltip } from
'@syncfusion/ej2-react-charts';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  const data: any[] = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category' };
  return <ChartComponent id="charts" primaryXAxis={primaryxAxis}
    title='Sales Analysis'>
    <Inject services={[ColumnSeries, Tooltip, LineSeries, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='month' yName='sales'
name='Sales' />
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Enable Legend

You can use legend for the chart by setting the `visible` property to true in [legendSettings](#) object and by injecting the `Legend` module into the services.

INDEX.JSX

```
import { Category, ChartComponent, ColumnSeries, Inject, Legend, LineSeries,
SeriesCollectionDirective, SeriesDirective, Tooltip } from '@syncfusion/ej2-
react-charts';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    const data = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    const legendSettings = { visible: true };
    const primaryxAxis = { valueType: 'Category' };
    return <ChartComponent id="charts" primaryXAxis={primaryxAxis}
legendSettings={legendSettings}>
        <Inject services={[ColumnSeries, Tooltip, Legend, LineSeries,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='month' yName='sales'
name='Sales' />
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import { AxisModel, Category, ChartComponent, ColumnSeries, Inject, Legend,
LegendSeriesModel, LineSeries, SeriesCollectionDirective, SeriesDirective,
Tooltip } from '@syncfusion/ej2-react-charts';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    const data: any[] = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    const legendSettings: LegendSeriesModel = { visible: true };
    const primaryxAxis: AxisModel = { valueType: 'Category' };
    return <ChartComponent id="charts" primaryXAxis={primaryxAxis}
legendSettings={legendSettings}>
        <Inject services={[ColumnSeries, Tooltip, Legend, LineSeries, Category]}
/>
        <SeriesCollectionDirective>
```

```

        <SeriesDirective dataSource={data} xName='month' yName='sales'
name='Sales' />
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Add Data Label

You can add data labels to improve the readability of the chart. This can be achieved by setting the visible property to true in the `dataLabel` object and by injecting `DataLabel` module into the services.

INDEX.JSX

```

import { Category, ChartComponent, ColumnSeries, DataLabel, Inject, Legend,
LineSeries, SeriesCollectionDirective, SeriesDirective, Tooltip } from
'@syncfusion/ej2-react-charts';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    const data = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    const primaryyAxis = { labelFormat: '${value}K' };
    const primaryxAxis = { valueType: 'Category' };
    const legendSettings = { visible: true };
    const marker = { dataLabel: { visible: true } };
    return <ChartComponent id="charts" primaryXAxis={primaryxAxis}
legendSettings={legendSettings} primaryYAxis={primaryyAxis}>
        <Inject services={[ColumnSeries, DataLabel, Tooltip, Legend, LineSeries,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='month' yName='sales'
name='Sales' marker={marker}/>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import {
    AxisModel, Category, ChartComponent, ColumnSeries, DataLabel, Inject,
    Legend,
    LegendSeriesModel, LineSeries, SeriesCollectionDirective,
    SeriesDirective, Tooltip
} from '@syncfusion/ej2-react-charts';
import * as React from "react";

```

```
import * as ReactDOM from "react-dom";
function App() {
  const data: any[] = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const primaryyAxis: AxisModel = { labelFormat: '${value}K' };
  const primaryxAxis: AxisModel = { valueType: 'Category' };
  const legendSettings: LegendSeriesModel = { visible: true };
  const marker = { dataLabel: { visible: true } };
  return <ChartComponent id="charts" primaryXAxis={primaryxAxis}
    legendSettings={legendSettings} primaryYAxis={primaryyAxis}>
    <Inject services={[ColumnSeries, DataLabel, Tooltip, Legend, LineSeries,
    Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='month' yName='sales'
name='Sales' marker={marker} />
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Enable Tooltip

The tooltip is useful when you cannot display information by using the data labels due to space constraints. You can enable tooltip by setting the enable property as true in [tooltip](#) object and by injecting [Tooltip](#) module into the [services](#).

INDEX.JSX

```
import { Category, ChartComponent, ColumnSeries, DataLabel, Inject, Legend,
LineSeries, SeriesCollectionDirective, SeriesDirective, Tooltip } from
'@syncfusion/ej2-react-charts';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  const data = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const tooltip = { enable: true, shared: false };
  const primaryyAxis = { labelFormat: '${value}K' };
  const primarxyAxis = { valueType: 'Category' };
  const legendSettings = { visible: true };
  const marker = { dataLabel: { visible: true } };
  return <ChartComponent id="charts" primaryXAxis={primarxyAxis}
    legendSettings={legendSettings} primaryYAxis={primaryyAxis}
    tooltip={tooltip}>
```

```

    <Inject services={[ColumnSeries, DataLabel, Tooltip, Legend, LineSeries,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='month' yName='sales'
name='Sales' marker={marker}/>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import {
  AxisModel, Category, ChartComponent, ColumnSeries, DataLabel, Inject,
  Legend, LegendSeriesModel, LineSeries, SeriesCollectionDirective,
  SeriesDirective, Tooltip, TooltipSettingsModel
} from '@syncfusion/ej2-react-charts';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  const data: any[] = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const tooltip: TooltipSettingsModel = { enable: true, shared: false }
  const primaryyAxis: AxisModel = { labelFormat: '${value}K' }
  const primarxyAxis: AxisModel = { valueType: 'Category' }
  const legendSettings: LegendSeriesModel = { visible: true }
  const marker = { dataLabel: { visible: true } };
  return <ChartComponent id="charts" primaryXAxis={primarxyAxis}
legendSettings={legendSettings}
  primaryYAxis={primaryyAxis} tooltip={tooltip}>
    <Inject services={[ColumnSeries, DataLabel, Tooltip, Legend, LineSeries,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='month' yName='sales'
name='Sales' marker={marker} />
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

You can refer to our [React Charts](#) feature tour page for its groundbreaking feature representations. You can also explore our [React Charts example](#) that shows various chart types and how to represent time-dependent data, showing trends in data at equal intervals.

Creating a Next.js Application Using Syncfusion React Components

This section provides a step-by-step guide for setting up a Next.js application and integrating the Syncfusion React Chart component.

What is Next.js?

[Next.js](#) is a React framework that makes it easy to build fast, SEO-friendly, and user-friendly web applications. It provides features such as server-side rendering, automatic code splitting, routing, and API routes, making it an excellent choice for building modern web applications.

Prerequisites

Before getting started with the Next.js application, ensure the following prerequisites are met:

- [Node.js 16.8](#) or later.
- The application is compatible with macOS, Windows, and Linux operating systems.

Create a Next.js application

To create a new **Next.js** application, use one of the commands that are specific to either NPM or Yarn.

NPM

```
npx create-next-app@latest
```

YARN

```
yarn create next-app
```

Using one of the above commands will lead you to set up additional configurations for the project as below:

1. Define the project name: Users can specify the name of the project directly. Let's specify the name of the project as **ej2-nextjs-chart**.

CMD

```
√ What is your project named? » ej2-nextjs-chart
```

2. Select the required packages.

CMD

```
√ What is your project named? ... ej2-nextjs-chart
√ Would you like to use TypeScript? ... No / `Yes`
√ Would you like to use ESLint? ... No / `Yes`
√ Would you like to use Tailwind CSS? ... `No` / Yes
√ Would you like to use `src/` directory? ... No / `Yes`
√ Would you like to use App Router? (recommended) ... No / `Yes`
√ Would you like to customize the default import alias? ... `No` / Yes
Creating a new Next.js app in D:\ej2-nextjs-chart.
```

3. Once complete the above mentioned steps to create **ej2-nextjs-chart**, navigate to the directory using the below command:

CMD

```
cd ej2-nextjs-chart
```

The application is ready to run with default settings. Now, let's add Syncfusion components to the project.

Install Syncfusion React packages

Syncfusion React component packages are available at [npmjs.com](https://www.npmjs.com). To use Syncfusion React components in the project, install the corresponding npm package.

Here, the [React Chart component](#) is used in the project. To install the React Chart component, use the following command:

NPM

```
npm install @syncfusion/ej2-react-charts --save
```

YARN

```
yarn add @syncfusion/ej2-react-charts
```

Add Syncfusion React component

Follow the below steps to add the React Chart component to the Next.js project:

1. Before adding the Chart component to your markup, import the Chart component in the **src/app/page.tsx** file.

PAGE.TSX

```
'use client'
import {
  AxisModel, Category, ChartComponent, ColumnSeries, DataLabel, Inject,
  Legend, LegendSeriesModel, LineSeries, SeriesCollectionDirective,
  SeriesDirective, Tooltip, TooltipSettingsModel
} from '@syncfusion/ej2-react-charts';
```

2. Then, define the Chart component in the **src/app/page.tsx** file, as shown below:

PAGE.TSX

```
'use client'
import {
  AxisModel, Category, ChartComponent, ColumnSeries, DataLabel, Inject,
  Legend, LegendSeriesModel, LineSeries, SeriesCollectionDirective,
  SeriesDirective, Tooltip, TooltipSettingsModel
} from '@syncfusion/ej2-react-charts';
export default function Home() {
  const data: any[] = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const tooltip: TooltipSettingsModel = { enable: true, shared: false }
```

```
const primaryyAxis: AxisModel = { labelFormat: '${value}K' }
const primarxyAxis: AxisModel = { valueType: 'Category' }
const legendSettings: LegendSeriesModel = { visible: true }
const marker = { dataLabel: { visible: true } };
return (
  <>
  <h2>Syncfusion React Chart Component</h2>
  <ChartComponent id="charts" primaryXAxis={primarxyAxis}
  legendSettings={legendSettings}
  primaryYAxis={primaryyAxis} tooltip={tooltip}>
  <Inject services={[ColumnSeries, DataLabel, Tooltip, Legend, LineSeries,
  Category]} />
  <SeriesCollectionDirective>
  <SeriesDirective dataSource={data} xName='month' yName='sales' name='Sales'
  marker={marker} />
  </SeriesCollectionDirective>
  </ChartComponent>
  </>
  )
}
```

Run the application

To run the application, use the following command:

NPM

```
npm run dev
```

YARN

```
yarn run dev
```

To learn more about the functionality of the Chart component, refer to the [documentation](#).

[View the NEXT.js Chart sample in the GitHub repository.](#)

Getting Started with the React Chart Component in the Preact Framework

This article provides a step-by-step guide for setting up a [Preact](#) project and integrating the Syncfusion React Chart component.

Preact is a fast and lightweight JavaScript library for building user interfaces. It's often used as an alternative to larger frameworks like React. The key difference is that Preact is designed to be smaller in size and faster in performance, making it a good choice for projects where file size and load times are critical factors.

Prerequisites

[System requirements for Syncfusion React UI components](#)

Set up the Preact project

To create a new **Preact** project, use one of the commands that are specific to either NPM or Yarn.

```
`bash
```

```
npm init preact
```

```
`
```

```
or
```

```
`bash
```

```
yarn init preact
```

```
`
```

Using one of the above commands will lead you to set up additional configurations for the project, as below:

1\ Define the project name: We can specify the name of the project directly. Let's specify the name of the project as `my-project` for this article.

```
`bash
```

```
T Preact - Fast 3kB alternative to React with the same modern API
```

```
|
```

- Project directory:

```
| my-project
```

```
—
```

```
`
```

2\ Choose `JavaScript` as the framework variant to build this Preact project using JavaScript and React.

```
`bash
```

```
T Preact - Fast 3kB alternative to React with the same modern API
```

```
|
```

- Project language:

```
| > JavaScript
```

```
| TypeScript
```

```
—
```

```
`
```

3\ Then configure the project as below for this article.

```
`bash
```

```
T Preact - Fast 3kB alternative to React with the same modern API
```

```
|
```

- Use router?

```
| Yes / > No
```


—
|

- Prerender app (SSG)?

| Yes / > No

—
|

- Use ESLint?

| Yes / > No

—
,

5\ Upon completing the aforementioned steps to create `my-project`, run the following command to jump into the project directory:

```
`bash
cd my-project
`
```

Now that `my-project` is ready to run with default settings, let's add Syncfusion components to the project.

Add the Syncfusion React packages

Syncfusion React component packages are available at [npmjs.com](https://www.npmjs.com). To use Syncfusion React components in the project, install the corresponding npm package.

This article uses the [React Chart component](#) as an example. To use the React Chart component in the project, the `@syncfusion/ej2-react-charts` package needs to be installed using the following command:

```
`bash
npm install @syncfusion/ej2-react-charts --save
`
```

or

```
`bash
yarn add @syncfusion/ej2-react-charts
`
```

Add the Syncfusion React component

Follow the below steps to add the React Chart component to the Vite project:

1\ Before adding the Chart component to your markup, import the Chart component in the **src/index.jsx** file.

~/SRC/INDEX.JSX

```
import { Category, ChartComponent, ColumnSeries, Inject, LineSeries,
SeriesCollectionDirective, SeriesDirective, Tooltip } from '@syncfusion/ej2-
react-charts';
```

2\ Then, define the Chart component in the **src/index.jsx** file, as shown below:

~/SRC/INDEX.JSX

```
import { render } from 'preact';
import { Category, ChartComponent, ColumnSeries, Inject, LineSeries,
SeriesCollectionDirective, SeriesDirective, Tooltip } from '@syncfusion/ej2-
react-charts';
export function App() {
  const data = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const primaryxAxis = { valueType: 'Category' };
  return (
    <ChartComponent id="charts" primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, Tooltip, LineSeries, Category]}/>
    <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='month' yName='sales'
    name='Sales' />
    </SeriesCollectionDirective>
    </ChartComponent>
  );
}
render(<App />, document.getElementById('app'));
```

Run the project

To run the project, use the following command:

```
`bash
```

```
npm run dev
```

```
,
```

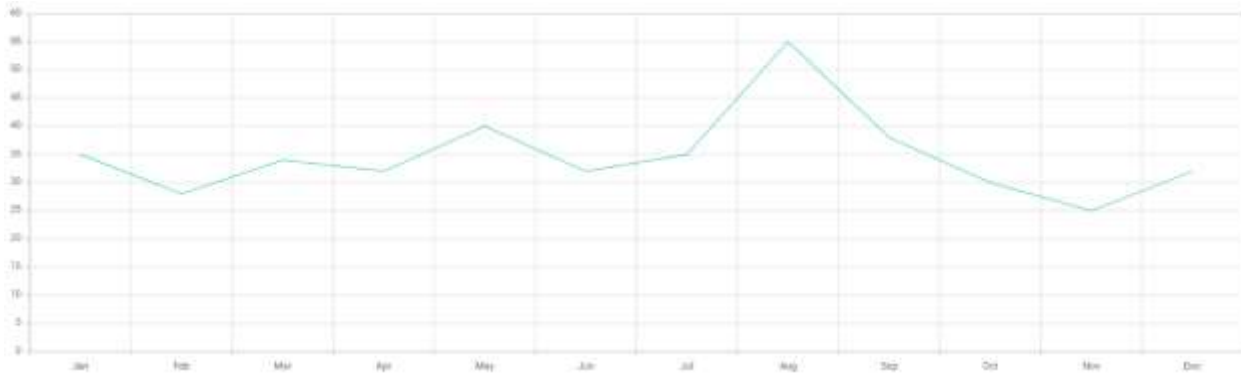
or

```
`bash
```

```
yarn run dev
```

```
,
```

The output will appear as follows:



See also

[Getting Started with the Syncfusion React UI Component](#)

<!-- markdownlint-disable MD036 -->

Working with data in React Chart component

Chart can visualise data bound from local or remote data.

Local Data

You can bind a simple JSON data to the chart using [dataSource](#) property in series. Now map the fields in JSON to [xName](#) and [yName](#) properties.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { Category, ChartComponent, ColumnSeries, Inject, Legend, LineSeries,
SeriesCollectionDirective, SeriesDirective } from '@syncfusion/ej2-react-charts';
function App() {
    const data = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    const primaryXAxis = { valueType: 'Category' };
    return <ChartComponent id='charts' primaryXAxis={primaryXAxis}>
        <Inject services={[ColumnSeries, Legend, LineSeries, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='month' type='Column'
yName='sales' name='Sales' />
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
```

```
import * as ReactDOM from "react-dom";
import { AxisModel, Category, ChartComponent, ColumnSeries, Inject, Legend,
LineSeries,
SeriesCollectionDirective, SeriesDirective } from '@syncfusion/ej2-react-
charts';
function App() {
  const data: any[] = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const primaryXAxis: AxisModel = { valueType: 'Category' };
  return <ChartComponent id='charts' primaryXAxis={primaryXAxis}>
    <Inject services={[ColumnSeries, Legend, LineSeries, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='month' type='Column'
yName='sales' name='Sales' />
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Lazy loading

Lazy loading allows you to load data for chart on demand. Chart will fire the scrollEnd event, in that we can get the minimum and maximum range of the axis, based on this, we can upload the data to chart.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ScrollBar, Zoom, LineSeries, Tooltip, DateTime, Crosshair } from
"@syncfusion/ej2-react-charts";
import { Internationalization } from '@syncfusion/ej2-base';
function App() {
  let chart;
  let intl = new Internationalization();
  function GetDateTimeData(start, end) {
    let series1 = [];
    let date;
    let value = 30;
    let option = {
      skeleton: 'full',
      type: 'dateTime'
    };
    let dateParser = intl.getDateParser(option);
    let dateFormatter = intl.getDateFormat(option);
    for (let i = 0; start <= end; i++) {
      date = Date.parse(dateParser(dateFormatter(start)));
      if (Math.random() > .5) {
        value += (Math.random() * 10 - 5);
      }
    }
  }
}
```

```

    }
    else {
        value -= (Math.random() * 10 - 5);
    }
    if (value < 0) {
        value = getRandomInt(20, 40);
    }
    let point1 = { x: new Date(date), y: Math.round(value) };
    new Date(start.setDate(start.getDate() + 1));
    series1.push(point1);
}
return series1;
}
function getRandomInt(min, max) {
    return Math.floor(Math.random() * (max - min + 1)) + min;
}
return <ChartComponent id='charts' ref={chart => chart = chart}
primaryXAxis={{
    title: 'Day',
    valueType: 'DateTime',
    edgeLabelPlacement: 'Shift',
    skeleton: 'yMMM',
    skeletonType: 'Date',
    scrollbarSettings: {
        range: {
            minimum: new Date(2009, 0, 1),
            maximum: new Date(2014, 0, 1)
        },
        enable: true,
    }
}} primaryYAxis={{
    title: 'Server Load',
    labelFormat: '{value}MB'
}} crosshair={{ enable: true, lineType: 'Vertical' }} chartArea={{
border: { width: 0 } }} tooltip={{ enable: true }} legendSettings={{
visible: true }}
// scrollStart={this.scrollStart.bind(this)}
// scrollEnd={this.scrollEnd.bind(this)}
title='Network Load' height='450' width='100%'>
    <Inject services={[LineSeries, DateTime,
Tooltip, ScrollBar, Zoom, Crosshair]}/>
    <SeriesCollectionDirective>
        <SeriesDirective
dataSource={GetDateTimeData(new Date(2009, 0, 1), new Date(2009, 8, 1))}
xName='x' yName='y' type='Line' animation={{ enable: false }}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective,
Inject,
    ChartTheme, ScrollBar, Zoom, IScrollEventArgs, LineSeries, Tooltip,
    DateTime, ILoadedEventArgs, Chart, Crosshair, ColumnSeries } from
'@syncfusion/ej2-react-charts';
import { Internationalization } from '@syncfusion/ej2-base';
function App() {
    let chart: ChartComponent;
    let intl: Internationalization = new Internationalization();
    function GetDateTimeData(start: Date, end: Date): { x: Date, y: number
}[] {
        let series1: { x: Date, y: number }[] = [];
        let date: number;
        let value: number = 30;
        let option: DateFormatOptions = {
            skeleton: 'full',
            type: 'dateTime'
        };
        let dateParser: Function = intl.getDateParser(option);
        let dateFormatter: Function = intl.getDateFormat(option);
        for (let i: number = 0; start <= end; i++) {
            date = Date.parse(dateParser(dateFormatter(start)));
            if (Math.random() > .5) {
                value += (Math.random() * 10 - 5);
            } else {
                value -= (Math.random() * 10 - 5);
            }
            if (value < 0) {
                value = getRandomInt(20, 40);
            }
            let point1: { x: Date, y: number } = { x: new Date(date), y:
Math.round(value) };
            new Date(start.setDate(start.getDate() + 1));
            series1.push(point1);
        }
        return series1;
    }
    function getRandomInt(min: number, max: number): number {
        return Math.floor(Math.random() * (max - min + 1)) + min;
    }

    return <ChartComponent id='charts'
        ref={chart => chart = chart}
        primaryXAxis={{
            title: 'Day',
            valueType: 'DateTime',
            edgeLabelPlacement: 'Shift',
            skeleton: 'yMMM',
            skeletonType: 'Date',
            scrollbarSettings: {
                range: {
                    minimum: new Date(2009, 0, 1),
                    maximum: new Date(2014, 0, 1)
                }
            }
        }

```

```

        enable: true,
      }
    }}
    primaryYAxis={{
      title: 'Server Load',
      labelFormat: '{value}MB'
    }}
    crosshair={{ enable: true, lineType: 'Vertical'
  }}

  chartArea={{ border: { width: 0 } }}
  tooltip={{ enable: true }}
  legendSettings={{ visible: true }}
  // scrollStart={this.scrollStart.bind(this)}
  // scrollEnd={this.scrollEnd.bind(this)}
  title='Network Load' height='450' width='100%' >
  <Inject services={[LineSeries, DateTime,
  Tooltip, ScrollBar, Zoom, Crosshair]} />
  <SeriesCollectionDirective>
    <SeriesDirective
  dataSource={GetDateTimeData(new Date(2009, 0, 1), new Date(2009, 8, 1))}
  xName='x' yName='y'
      type='Line' animation={{ enable: false
  }}>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% enddraw %}

```

Common DataSource

You can also bind a JSON data common to all series using [dataSource](#) property in chart.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Category, ChartComponent, ColumnSeries, Inject, Legend, LineSeries,
SeriesCollectionDirective, SeriesDirective } from '@syncfusion/ej2-react-
charts';
function App() {
  const chartData = [
    { month: 'Jan', sales: 35, sales1: 28 }, { month: 'Feb', sales: 28,
sales1: 35 },
    { month: 'Mar', sales: 34, sales1: 32 }, { month: 'Apr', sales: 32,
sales1: 34 },
    { month: 'May', sales: 40, sales1: 32 }, { month: 'Jun', sales: 32,
sales1: 40 },
    { month: 'Jul', sales: 35, sales1: 55 }, { month: 'Aug', sales: 55,
sales1: 35 },
    { month: 'Sep', sales: 38, sales1: 30 }, { month: 'Oct', sales: 30,
sales1: 38 },
    { month: 'Nov', sales: 25, sales1: 32 }, { month: 'Dec', sales: 32,
sales1: 25 }
  ];
}

```

```

const primaryxAxis = { valueType: 'Category' };
return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
dataSource={chartData}>
  <Inject services={[ColumnSeries, Legend, LineSeries, Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective xName='month' type='Column' yName='sales' />
    <SeriesDirective xName='month' type='Column' yName='sales1' />
  </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, Category, ChartComponent, ColumnSeries, Inject, Legend,
LineSeries,
SeriesCollectionDirective, SeriesDirective } from '@syncfusion/ej2-react-
charts';
function App() {
  const chartData: any[] = [
    { month: 'Jan', sales: 35, sales1: 28 }, { month: 'Feb', sales: 28,
sales1: 35 },
    { month: 'Mar', sales: 34, sales1: 32 }, { month: 'Apr', sales: 32,
sales1: 34 },
    { month: 'May', sales: 40, sales1: 32 }, { month: 'Jun', sales: 32,
sales1: 40 },
    { month: 'Jul', sales: 35, sales1: 55 }, { month: 'Aug', sales: 55,
sales1: 35 },
    { month: 'Sep', sales: 38, sales1: 30 }, { month: 'Oct', sales: 30,
sales1: 38 },
    { month: 'Nov', sales: 25, sales1: 32 }, { month: 'Dec', sales: 32,
sales1: 25 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
dataSource={chartData}>
    <Inject services={[ColumnSeries, Legend, LineSeries, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective xName='month' type='Column' yName='sales' />
      <SeriesDirective xName='month' type='Column' yName='sales1' />
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Remote Data

You can also bind remote data to the chart using **DataManager**. The DataManager requires minimal information like webservice URL, adaptor and crossDomain to interact with service endpoint properly.

Assign the instance of DataManager to the [dataSource](#) property in series and map the fields of data to [xName](#) and [yName](#) properties. You can also use the [query](#) property of the series to filter the data.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { DataManager, Query } from '@syncfusion/ej2-data';
import { Category, ChartComponent, ColumnSeries, Inject, Legend, LineSeries,
SeriesCollectionDirective, SeriesDirective } from '@syncfusion/ej2-react-
charts';
function App() {
  const dataManager = new DataManager({
    url: 'https://services.syncfusion.com/js/production/api/orders'
  });
  const query = new Query().take(5).where('Estimate', 'lessThan', 3, false);
  const primaryxAxis = { valueType: 'Category' };
  const primaryyAxis = { title: 'Freight rate in U.S. dollars' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title="Container freight rate">
    <Inject services={[ColumnSeries, Legend, Category, LineSeries]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={dataManager} xName='CustomerID'
type='Column' yName='Freight' query={query} />
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { DataManager, Query } from '@syncfusion/ej2-data';
import { AxisModel, Category, ChartComponent, ColumnSeries, Inject, Legend,
LineSeries, SeriesCollectionDirective, SeriesDirective } from
 '@syncfusion/ej2-react-charts';
function App() {
  const dataManager = new DataManager({
    url: 'https://services.syncfusion.com/js/production/api/orders'
  });
  const query = new Query().take(5).where('Estimate', 'lessThan', 3, false);
  const primaryxAxis: AxisModel = { valueType: 'Category' };
  const primaryyAxis: AxisModel = { title: 'Freight rate in U.S. dollars' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title="Container freight rate">
    <Inject services={[ColumnSeries, Legend, Category, LineSeries]} />
    <SeriesCollectionDirective>

```

```

        <SeriesDirective dataSource={dataManager} xName='CustomerID'
        type='Column' yName='Freight' query={query} />
      </SeriesCollectionDirective>
    </ChartComponent>
  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));
  {% enddraw %}

```

Empty points

The Data points that uses the `null` or `undefined` as value are considered as empty points. Empty data points are ignored and not plotted in the Chart. When the data is provided by using the points property, By using `emptyPointSettings` property in series, you can customize the empty point. Default `mode` of the empty point is `Gap`.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Category, ChartComponent, ColumnSeries, Inject, LineSeries,
SeriesCollectionDirective, SeriesDirective } from '@syncfusion/ej2-react-charts';
function App() {
  const data = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: null },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: undefined },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const primaryxAxis = { valueType: 'Category' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, LineSeries, ColumnSeries,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='month' type='Column'
yName='sales' name='Sales' />
      <SeriesDirective dataSource={data} xName='month' type='Line'
yName='sales' name='Sales' />
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, Category, ChartComponent, ColumnSeries, Inject,
LineSeries,
SeriesCollectionDirective, SeriesDirective } from '@syncfusion/ej2-react-charts';

```

```
function App() {
  const data: any[] = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: null },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: undefined },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, LineSeries, ColumnSeries, Category]}
  />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='month' type='Column'
yName='sales' name='Sales' />
      <SeriesDirective dataSource={data} xName='month' type='Line'
yName='sales' name='Sales' />
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Empty point color

Specific color for empty point can be set by `fill` property in `emptyPointSettings`.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { Category, ChartComponent, ColumnSeries, Inject, LineSeries,
SeriesCollectionDirective, SeriesDirective } from '@syncfusion/ej2-react-
charts';
function App() {
  const data = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: null },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: undefined },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const primaryxAxis = { valueType: 'Category' };
  const emptyPointSettings1 = { mode: 'Average', fill: 'pink' };
  const emptyPointSettings2 = { mode: 'Zero', fill: 'green' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, LineSeries, ColumnSeries,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='month' type='Column'
yName='sales' name='Sales' emptyPointSettings={emptyPointSettings1} />
      <SeriesDirective dataSource={data} xName='month' type='Line'
yName='sales' name='Sales' emptyPointSettings={emptyPointSettings2} />
    </SeriesCollectionDirective>
  </ChartComponent>;
}
```

```

}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, Category, ChartComponent, ColumnSeries,
EmptyPointSettingsModel,
Inject, LineSeries, SeriesCollectionDirective, SeriesDirective } from
'@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: null },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: undefined },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category' };
  const emptyPointSettings1: EmptyPointSettingsModel = { mode: 'Average',
fill: 'pink' };
  const emptyPointSettings2: EmptyPointSettingsModel = { mode: 'Zero', fill:
'green' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, LineSeries, ColumnSeries, Category]}
/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='month' type='Column'
yName='sales' name='Sales' emptyPointSettings={emptyPointSettings1} />
      <SeriesDirective dataSource={data} xName='month' type='Line'
yName='sales' name='Sales' emptyPointSettings={emptyPointSettings2} />
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Chart dimensions in React Chart component

Size for Container

Chart can render to its container size. You can set the size via inline or CSS as demonstrated below.

```
<div id="charts" style="width:650px; height:350px"></div>
```

```
`ts
```

```
import * as React from "react";
```

```
import * as ReactDOM from "react-dom";
```

```

import { ChartComponent, SeriesCollectionDirective, AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
return <ChartComponent id='charts'>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
`ts
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent } from '@syncfusion/ej2-react-charts';
function App() {
return <ChartComponent id='charts'>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
`

```

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Category, ChartComponent, ColumnSeries, DataLabel, Inject, Legend,
LineSeries, SeriesCollectionDirective, SeriesDirective, Tooltip } from
'@syncfusion/ej2-react-charts';
function App() {
  const data = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const primaryxAxis = { valueType: 'Category' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>

```

```

    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='month' yName='sales'
type='Column' name='Sales' />
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, Category, ChartComponent, ColumnSeries, DataLabel,
Inject, Legend, LineSeries, SeriesCollectionDirective, SeriesDirective,
Tooltip } from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='month' yName='sales'
type='Column' name='Sales' />
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Size for Chart

You can also set size for chart directly through [width](#) and [height](#) properties.

<!-- markdownlint-disable MD036 -->

In Pixel

<!-- markdownlint-disable MD036 -->

You can set the size of chart in pixel as demonstrated below.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Category, ChartComponent, ColumnSeries, DataLabel, Inject, Legend,
LineSeries, SeriesCollectionDirective, SeriesDirective, Tooltip } from
'@syncfusion/ej2-react-charts';
function App() {
    const data = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    const primaryxAxis = { valueType: 'Category' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
width='650' height='350'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='month' yName='sales'
type='Column' name='Sales' />
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, Category, ChartComponent, ColumnSeries, DataLabel,
Inject, Legend, LineSeries, SeriesCollectionDirective, SeriesDirective,
Tooltip } from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    const primaryxAxis: AxisModel = { valueType: 'Category' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis} width='650'
height='350'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='month' yName='sales'
type='Column' name='Sales' />
        </SeriesCollectionDirective>
    </ChartComponent>
};

```

```
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

In Percentage

By setting value in percentage, chart gets its dimension with respect to its container. For example, when the height is '50%', chart renders to half of the container height.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { Category, ChartComponent, ColumnSeries, DataLabel, Inject, Legend,
LineSeries, SeriesCollectionDirective, SeriesDirective, Tooltip } from
'@syncfusion/ej2-react-charts';
function App() {
    const data = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    const primaryxAxis = { valueType: 'Category' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
width='80%' height='90%'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='month' yName='sales'
type='Column' name='Sales' />
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, Category, ChartComponent, ColumnSeries, DataLabel,
Inject, Legend, LineSeries, SeriesCollectionDirective, SeriesDirective,
Tooltip } from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    const primaryxAxis: AxisModel = { valueType: 'Category' };
}
```



```

return <ChartComponent id='charts' primaryXAxis={primaryxAxis} width='80%'
height='90%'>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='month' yName='sales'
type='Column' name='Sales' />
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Note: When you do not specify the size, it takes [Link to the Video](#) as the height and window size as its width.

Category axis in React Chart component

Category axis are used to represent, the string values instead of numbers.

To get start quickly with React Category Axis, you can check out this video:

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category } from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'Category', title: 'Countries' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='country' yName='gold'
type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend, Category }
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
  const chartData: Object[] = [
    { country: "USA", gold: 50 },

```

```

    { country: "China", gold: 40 },
    { country: "Japan", gold: 70 },
    { country: "Australia", gold: 60 },
    { country: "France", gold: 50 },
    { country: "Germany", gold: 40 },
    { country: "Italy", gold: 40 },
    { country: "Sweden", gold: 30 }
  ];
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
    title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='country' yName='gold'
type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  };
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Note: To use category axis, we need to inject **Category** module into the **services** and set the [valueType](#) of axis to Category.

Labels Placement

By default, category labels are placed between the ticks in an axis, this can also be placed on ticks using [labelPlacement](#) property.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category } from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'Category', title: 'Countries',
labelPlacement: 'OnTicks' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='country' yName='gold'
type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";

```

```
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend, Category }
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryXAxis: AxisModel = { valueType: 'Category', title:
'Countries', labelPlacement: 'OnTicks' };
  const chartData: any[] = [
    { country: "USA", gold: 50 },
    { country: "China", gold: 40 },
    { country: "Japan", gold: 70 },
    { country: "Australia", gold: 60 },
    { country: "France", gold: 50 },
    { country: "Germany", gold: 40 },
    { country: "Italy", gold: 40 },
    { country: "Sweden", gold: 30 }
  ];
  return <ChartComponent id='charts' primaryXAxis={primaryXAxis}
    title='Olympic Medals' >
    <Inject services={[ColumnSeries, Legend, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='country' yName='gold'
type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  };
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Range

Range of the category axis can be customized using [minimum](#), [maximum](#) and [interval](#) property of the axis.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category } from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryXAxis = { valueType: 'Category', title: 'Countries',
interval: 2, minimum: 1, maximum: 5 };
  return <ChartComponent id='charts' primaryXAxis={primaryXAxis}
title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='country' yName='gold'
type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
  }
;
export default App;
```

```
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend, Category }
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryXAxis: AxisModel = { valueType: 'Category', title:
'Countries', interval: 2, minimum: 1, maximum: 5 };
    const chartData: any[] = [
        { country: "USA", gold: 50 },
        { country: "China", gold: 40 },
        { country: "Japan", gold: 70 },
        { country: "Australia", gold: 60 },
        { country: "France", gold: 50 },
        { country: "Germany", gold: 40 },
        { country: "Italy", gold: 40 },
        { country: "Sweden", gold: 30 }
    ];

    return <ChartComponent id='charts' primaryXAxis={primaryXAxis}
        title='Olympic Medals' >
        <Inject services={[ColumnSeries, Legend, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} xName='country' yName='gold'
type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Indexed category axis

Category axis also can be rendered based on the index values of data source. This can be achieved by defining the `isIndexed` property to `true` in the axis.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { Category, ChartComponent, ColumnSeries, Inject, Legend,
SeriesCollectionDirective, SeriesDirective } from '@syncfusion/ej2-react-
charts';
function App() {
    const data1 = [{ x: 'Myanmar', y: 7.3 }, { x: 'India', y: 7.9 },
        { x: 'Bangladesh', y: 6.8 }, { x: 'Cambodia', y: 7.0 }, { x:
'China', y: 6.9 }];
    const data2 = [{ x: 'Poland', y: 2.7 }, { x: 'Australia', y: 2.5 },
        { x: 'Singapore', y: 2.0 }, { x: 'Canada', y: 1.4 }, { x: 'Germany',
y: 1.8 }];
    const primaryXAxis = { valueType: 'Category', isIndexed: true };
}
```

```

    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
        <Inject services={[ColumnSeries, Legend,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data1} xName='x'
yName='y' type='Column' />
            <SeriesDirective dataSource={data1} xName='x'
yName='y' type='Column' />
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, Category, ChartComponent, ColumnSeries, Inject, Legend,
SeriesCollectionDirective, SeriesDirective } from '@syncfusion/ej2-react-
charts';
function App() {
    const data1: any[] = [{ x: 'Myanmar', y: 7.3 }, { x: 'India', y: 7.9 },
{ x: 'Bangladesh', y: 6.8 }, { x: 'Cambodia', y:
7.0 }, { x: 'China', y: 6.9 }];
    const data2: any[] = [{ x: 'Poland', y: 2.7 }, { x: 'Australia', y: 2.5
},
{ x: 'Singapore', y: 2.0 }, { x: 'Canada', y: 1.4
}, { x: 'Germany', y: 1.8 }];
    const primaryxAxis: AxisModel= { valueType: 'Category', isIndexed:
true };
    return <ChartComponent id='charts'
primaryXAxis={ primaryxAxis }>
        <Inject services={[ColumnSeries, Legend,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource ={data1} xName='x'
yName='y' type='Column' />
            <SeriesDirective dataSource ={data1} xName='x'
yName='y' type='Column' />
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

<!-- markdownlint-disable MD036 -->

Numeric axis in React Chart component

You can use numeric axis to represent numeric values of data in chart. By default, the `valueType` of an axis is `Double`.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, AreaSeries } from
'@syncfusion/ej2-react-charts';
import { numericData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Double', title: 'Overs' };
    const primaryyAxis = { title: 'Runs' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='England - Run Rate'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Area'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
AreaSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Double', title: 'Overs' };
    const primaryyAxis: AxisModel = { title: 'Runs' };
    const numericData: any[] = [
        { x: 1, y: 7 }, { x: 2, y: 1 }, { x: 3, y: 1 },
        { x: 4, y: 14 }, { x: 5, y: 1 }, { x: 6, y: 10 },
        { x: 7, y: 8 }, { x: 8, y: 6 }, { x: 9, y: 10 },
        { x: 10, y: 10 }, { x: 11, y: 16 }, { x: 12, y: 6 },
        { x: 13, y: 14 }, { x: 14, y: 7 }, { x: 15, y: 5 },
        { x: 16, y: 2 }, { x: 17, y: 14 }, { x: 18, y: 7 },
        { x: 19, y: 7 }, { x: 20, y: 10 }
    ];
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis}
title='England - Run Rate'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Area'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>

```

```
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Range

Range for an axis, will be calculated automatically based on the provided data, you can also customize the range of the axis using [minimum](#), [maximum](#) and [interval](#) property of the axis.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, AreaSeries } from
'@syncfusion/ej2-react-charts';
import { numericData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Double', title: 'Overs', minimum: 1,
maximum: 20, interval: 5 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
title='England - Run Rate'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Area'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
AreaSeries, Selection }
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Double', title: 'Overs',
minimum: 1, maximum: 20, interval: 5 };
    const numericData: any[] = [
        { x: 1, y: 7 }, { x: 2, y: 1 }, { x: 3, y: 1 },
        { x: 4, y: 14 }, { x: 5, y: 1 }, { x: 6, y: 10 },
        { x: 7, y: 8 }, { x: 8, y: 6 }, { x: 9, y: 10 },
        { x: 10, y: 10 }, { x: 11, y: 16 }, { x: 12, y: 6 },
        { x: 13, y: 14 }, { x: 14, y: 7 }, { x: 15, y: 5 },
        { x: 16, y: 2 }, { x: 17, y: 14 }, { x: 18, y: 7 },
        { x: 19, y: 7 }, { x: 20, y: 10 }
    ];
};
```

```

return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  title='England - Run Rate'>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Area'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Range Padding

Padding can be applied to the minimum and maximum extremes of the axis range by using the [rangePadding](#) property. Numeric axis supports following types of padding.

- None
- Round
- Additional
- Normal
- Auto

Numeric - None

When the [rangePadding](#) is set to **None**, minimum and maximum of an axis is based on the data.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, AreaSeries } from
'@syncfusion/ej2-react-charts';
import { numericData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'Double', title: 'Overs',
rangePadding: 'None' };
  const primaryyAxis = { title: 'Runs' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='England - Run Rate'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Area'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;

```



```
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
AreaSeries, Selection }
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Double', title: 'Overs',
rangePadding: 'None' };
    const primaryyAxis: AxisModel = { title: 'Runs' };
    const numericData: any[] = [
        { x: 1, y: 7 }, { x: 2, y: 1 }, { x: 3, y: 1 },
        { x: 4, y: 14 }, { x: 5, y: 1 }, { x: 6, y: 10 },
        { x: 7, y: 8 }, { x: 8, y: 6 }, { x: 9, y: 10 },
        { x: 10, y: 10 }, { x: 11, y: 16 }, { x: 12, y: 6 },
        { x: 13, y: 14 }, { x: 14, y: 7 }, { x: 15, y: 5 },
        { x: 16, y: 2 }, { x: 17, y: 14 }, { x: 18, y: 7 },
        { x: 19, y: 7 }, { x: 20, y: 10 }
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='England - Run Rate'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Area'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Numeric - Round

When the [rangePadding](#) is set to **Round**, minimum and maximum will be rounded to the nearest possible value divisible by interval. For example, when the minimum is 3.5 and the interval is 1, then the minimum will be rounded to 3.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, AreaSeries } from
 '@syncfusion/ej2-react-charts';
import { numericData } from 'datasource.ts';
function App() {
```

```

    const primaryxAxis = { valueType: 'Double', title: 'Overs',
rangePadding: 'Round' };
    const primaryyAxis = { title: 'Runs' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='England - Run Rate'>
      <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]}>
        <SeriesCollectionDirective>
          <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Area'>
            </SeriesDirective>
          </SeriesCollectionDirective>
        </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
AreaSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'Double', title: 'Overs',
rangePadding: 'Round' };
  const primaryyAxis: AxisModel = { title: 'Runs' };
  const numericData: any[] = [
    { x: 1, y: 7 }, { x: 2, y: 1 }, { x: 3, y: 1 },
    { x: 4, y: 14 }, { x: 5, y: 1 }, { x: 6, y: 10 },
    { x: 7, y: 8 }, { x: 8, y: 6 }, { x: 9, y: 10 },
    { x: 10, y: 10 }, { x: 11, y: 16 }, { x: 12, y: 6 },
    { x: 13, y: 14 }, { x: 14, y: 7 }, { x: 15, y: 5 },
    { x: 16, y: 2 }, { x: 17, y: 14 }, { x: 18, y: 7 },
    { x: 19, y: 7 }, { x: 20, y: 10 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='England - Run Rate'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Area'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));

```

Numeric - Additional

When the [rangePadding](#) is set to **Additional**, interval of an axis will be padded to the minimum and maximum of the axis.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, AreaSeries } from
'@syncfusion/ej2-react-charts';
import { numericData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Double', title: 'Overs',
rangePadding: 'Additional' };
    const primaryyAxis = { title: 'Runs' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='England - Run Rate'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Area'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
AreaSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Double', title: 'Overs',
rangePadding: 'Additional' };
    const primaryyAxis: AxisModel = { title: 'Runs' };
    const numericData: any[] = [
        { x: 1, y: 7 }, { x: 2, y: 1 }, { x: 3, y: 1 },
        { x: 4, y: 14 }, { x: 5, y: 1 }, { x: 6, y: 10 },
        { x: 7, y: 8 }, { x: 8, y: 6 }, { x: 9, y: 10 },
        { x: 10, y: 10 }, { x: 11, y: 16 }, { x: 12, y: 6 },
        { x: 13, y: 14 }, { x: 14, y: 7 }, { x: 15, y: 5 },
        { x: 16, y: 2 }, { x: 17, y: 14 }, { x: 18, y: 7 },
        { x: 19, y: 7 }, { x: 20, y: 10 }
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
```

```

        title='England - Run Rate'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Area'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Numeric - Normal

When the [rangePadding](#) is set to **Normal**, padding is applied to the axis based on default range calculation.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, AreaSeries } from
'@syncfusion/ej2-react-charts';
import { numericData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Double', title: 'Overs',
rangePadding: 'Normal' };
    const primaryyAxis = { title: 'Runs' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='England - Run Rate'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Area'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
AreaSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {

```

```

const primaryxAxis: AxisModel = { valueType: 'Double', title: 'Overs',
rangePadding: 'Normal' };
const primaryyAxis: AxisModel = { title: 'Runs' };
const numericData: any[] = [
  { x: 1, y: 7 }, { x: 2, y: 1 }, { x: 3, y: 1 },
  { x: 4, y: 14 }, { x: 5, y: 1 }, { x: 6, y: 10 },
  { x: 7, y: 8 }, { x: 8, y: 6 }, { x: 9, y: 10 },
  { x: 10, y: 10 }, { x: 11, y: 16 }, { x: 12, y: 6 },
  { x: 13, y: 14 }, { x: 14, y: 7 }, { x: 15, y: 5 },
  { x: 16, y: 2 }, { x: 17, y: 14 }, { x: 18, y: 7 },
  { x: 19, y: 7 }, { x: 20, y: 10 }
];
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  title='England - Run Rate'>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Area'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Numeric - Auto

When the [rangePadding](#) is set to **Auto**, horizontal numeric axis takes None as padding calculation, while the vertical numeric axis takes Normal as padding calculation.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, AreaSeries } from
'@syncfusion/ej2-react-charts';
import { numericData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'Double', title: 'Overs',
rangePadding: 'Auto' };
  const primaryyAxis = { title: 'Runs' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='England - Run Rate'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Area'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}

```

```
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
AreaSeries, Selection }
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Double', title: 'Overs',
rangePadding: 'Auto' };
    const primaryyAxis: AxisModel = { title: 'Runs' };
    const numericData: any[] = [
        { x: 1, y: 7 }, { x: 2, y: 1 }, { x: 3, y: 1 },
        { x: 4, y: 14 }, { x: 5, y: 1 }, { x: 6, y: 10 },
        { x: 7, y: 8 }, { x: 8, y: 6 }, { x: 9, y: 10 },
        { x: 10, y: 10 }, { x: 11, y: 16 }, { x: 12, y: 6 },
        { x: 13, y: 14 }, { x: 14, y: 7 }, { x: 15, y: 5 },
        { x: 16, y: 2 }, { x: 17, y: 14 }, { x: 18, y: 7 },
        { x: 19, y: 7 }, { x: 20, y: 10 }
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='England - Run Rate'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Area'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Label Format

Numeric Label Format

Numeric labels can be formatted by using the [labelFormat](#) property. Numeric labels supports all globalize format.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, AreaSeries } from
'@syncfusion/ej2-react-charts';
import { numericData } from 'datasource.ts';
```

```
function App() {
  const primaryxAxis = { title: 'Runs', labelFormat: 'c' };
  return <ChartComponent id='charts' primaryYAxis={primaryxAxis}
title='England - Run Rate'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Area'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
AreaSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { title: 'Runs', labelFormat: 'c' };
  const numericData: any[] = [
    { x: 1, y: 7 }, { x: 2, y: 1 }, { x: 3, y: 1 },
    { x: 4, y: 14 }, { x: 5, y: 1 }, { x: 6, y: 10 },
    { x: 7, y: 8 }, { x: 8, y: 6 }, { x: 9, y: 10 },
    { x: 10, y: 10 }, { x: 11, y: 16 }, { x: 12, y: 6 },
    { x: 13, y: 14 }, { x: 14, y: 7 }, { x: 15, y: 5 },
    { x: 16, y: 2 }, { x: 17, y: 14 }, { x: 18, y: 7 },
    { x: 19, y: 7 }, { x: 20, y: 10 }
  ];
  return <ChartComponent id='charts'
    primaryYAxis={primaryxAxis}
    title='England - Run Rate'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Area'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));
```

The following table describes the result of applying some commonly used label formats on numeric values.

<!-- markdownlint-disable MD033 -->

Label Value	Label Format property value	Result	Description
1000	n1	1000.0	The Number is rounded to 1 decimal place
1000	n2	1000.00	The Number is rounded to 2 decimal place
1000	n3	1000.000	The Number is rounded to 3 decimal place
0.01	p1	1.0%	The Number is converted to percentage with 1 decimal place
0.01	p2	1.00%	The Number is converted to percentage with 2 decimal place
0.01	p3	1.000%	The Number is converted to percentage with 3 decimal place
1000	c1	\$1000.0	The Currency symbol is appended to number and number is rounded to 1 decimal place
1000	c2	\$1000.00	The Currency symbol is appended to number and number is rounded to 2 decimal place

GroupingSeparator

To separate groups of thousands, use [useGroupingSeparator](#) property in chart.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, AreaSeries } from
'@syncfusion/ej2-react-charts';
import { groupingData } from 'datasource.ts';
function App() {
    const primaryyAxis = { title: 'Runs' };
    return <ChartComponent id='charts' primaryYAxis={primaryyAxis}
title='England - Sales Rate' useGroupingSeparator={true}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={groupingData} xName='x' yName='y'
name='England' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX


```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
AreaSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryyAxis: AxisModel = { title: 'Runs' };
    const groupingData: any[] = [
        { x: 10, y: 7000 }, { x: 20, y: 1000 }, { x: 30, y: 12000 }, { x: 40, y:
14000 }, { x: 50, y: 11000 }, { x: 60, y: 5000 },
        { x: 70, y: 7300 }, { x: 80, y: 9000 }, { x: 90, y: 12000 }, { x: 100,
y: 14000 }, { x: 110, y: 11000 }, { x: 120, y: 5000 },
    ];
    return <ChartComponent id='charts'
        primaryYAxis={primaryyAxis}
        title='England - Sales Rate' useGroupingSeparator={true}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={groupingData} xName='x' yName='y'
name='England' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
    };
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));
```

Custom Label Format

Axis also supports custom label format using placeholder like {value}°C, in which the value represent the axis label e.g 20°C.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, AreaSeries } from
'@syncfusion/ej2-react-charts';
import { numericData } from 'datasource.ts';
function App() {
    const primaryyAxis = { labelFormat: '${value}K' };
    return <ChartComponent id='charts' primaryYAxis={primaryyAxis}
title='England - Run Rate'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
```

```
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
AreaSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryyAxis: AxisModel = { labelFormat: '${value}K' };
  const numericData: any[] = [
    { x: 1, y: 7 }, { x: 2, y: 1 }, { x: 3, y: 1 },
    { x: 4, y: 14 }, { x: 5, y: 1 }, { x: 6, y: 10 },
    { x: 7, y: 8 }, { x: 8, y: 6 }, { x: 9, y: 10 },
    { x: 10, y: 10 }, { x: 11, y: 16 }, { x: 12, y: 6 },
    { x: 13, y: 14 }, { x: 14, y: 7 }, { x: 15, y: 5 },
    { x: 16, y: 2 }, { x: 17, y: 14 }, { x: 18, y: 7 },
    { x: 19, y: 7 }, { x: 20, y: 10 }
  ];
  return <ChartComponent id='charts'
    primaryYAxis={primaryyAxis}
    title='England - Run Rate'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
AreaSeries, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={numericData} xName='x' yName='y'
name='England' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

<!-- markdownlint-disable MD036 -->

Date time axis in React Chart component

DateTime Axis

Date time axis uses date time scale and displays the date time values as axis labels in the specified format.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Tooltip, DataLabel, LineSeries } from
 '@syncfusion/ej2-react-charts';
import { dateTimeData } from 'datasource.ts';
function App() {
```

```

    const primaryxAxis = { valueType: 'DateTime', title: 'Sales Across
Years', labelFormat: 'yMMM' };
    const primaryyAxis = { title: 'Sales Amount in millions(USD)' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Average Sales Comparison'>
      <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, DateTime]}>
        <SeriesCollectionDirective>
          <SeriesDirective dataSource={dateTimeData} xName='x' yName='y'
name='Sales' type='Line'>
            </SeriesDirective>
          </SeriesCollectionDirective>
        </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime', title: 'Sales
Across Years', labelFormat: 'yMMM' };
  const primaryyAxis: AxisModel = { title: 'Sales Amount in millions(USD)'
};
  const dateTimeData: any[] = [
    { x: new Date(2000, 6, 11), y: 10 }, { x: new Date(2002, 3, 7), y: 30 },
    { x: new Date(2004, 3, 6), y: 15 }, { x: new Date(2006, 3, 30), y: 65 },
    { x: new Date(2008, 3, 8), y: 90 }, { x: new Date(2010, 3, 8), y: 85 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Average Sales Comparison'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={dateTimeData} xName='x' yName='y'
name='Sales' type='Line'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));
}

```

Note: To use datetime axis, we need to inject **DateTime** module into the **services** and

set the [valueType](#) of axis to DateTime.

DateTimeCategory Axis

Date-time category axis is used to display the date-time values with non-linear intervals. For example, the business days alone have been depicted in a week here.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTimeCategory, Tooltip, DataLabel, LineSeries }
from '@syncfusion/ej2-react-charts';
import { dateTimeCategoryData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'DateTimeCategory', skeleton: 'Ed' };
    const primaryyAxis = { title: 'Sales Amount in millions(USD)' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Average Sales Comparison'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, DateTimeCategory]}>
            <SeriesCollectionDirective>
                <SeriesDirective dataSource={dateTimeCategoryData} xName='x'
yName='y' name='Sales' type='Line'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTimeCategory, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTimeCategory', skeleton:
'Ed' };
    const primaryyAxis: AxisModel = { title: 'Sales Amount in millions(USD)'
};
    const dateTimeCategoryData: any[] = [{ x: new Date(2017, 11, 20), y: 21 },
{ x: new Date(2017, 11, 21), y: 24 },
    { x: new Date(2017, 11, 22), y: 24 }, { x: new Date(2017, 11, 26), y: 70
},
    { x: new Date(2017, 11, 27), y: 75 }, { x: new Date(2018, 0, 2), y: 82
},
    { x: new Date(2018, 0, 3), y: 53 }, { x: new Date(2018, 0, 4), y: 54 },
    { x: new Date(2018, 0, 5), y: 53 }, { x: new Date(2018, 0, 8), y: 45 }
];
    return <ChartComponent id='charts'>
```

```

    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Average Sales Comparison'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, DateTimeCategory]} />
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={dateTimeCategoryData} xName='x'
yName='y' name='Sales' type='Line'>
        </SeriesDirective>
    </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Note: To use `dateTimeCategory` axis, we need to inject `DateTimeCategory` module into the `services` method and

set the [valueType](#) of axis to `DateTimeCategory`.

Range

Range for an axis, will be calculated automatically based on the provided data, you can also customize the range of the axis using [minimum](#), [maximum](#) and [interval](#) property of the axis.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Tooltip, DataLabel, LineSeries } from
'@syncfusion/ej2-react-charts';
import { dateTimeData } from 'datasource.ts';
function App() {
    const primaryxAxis = {
        valueType: 'DateTime', title: 'Sales Across Years', labelFormat:
'yMMM',
        minimum: new Date(2000, 6, 1), maximum: new Date(2010, 6, 1)
    };
    const primaryyAxis = { title: 'Sales Amount in millions(USD)' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Average Sales Comparison'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, DateTime]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={dateTimeData} xName='x' yName='y'
name='Sales' type='Line'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = {
    valueType: 'DateTime', title: 'Sales Across Years', labelFormat: 'yMMM',
    minimum: new Date(2000, 6, 1), maximum: new Date(2010, 6, 1)
  };
  const primaryyAxis: AxisModel = { title: 'Sales Amount in millions(USD)' };
};
const dateTimeData: any[] = [
  { x: new Date(2000, 6, 11), y: 10 }, { x: new Date(2002, 3, 7), y: 30 },
  { x: new Date(2004, 3, 6), y: 15 }, { x: new Date(2006, 3, 30), y: 65 },
  { x: new Date(2008, 3, 8), y: 90 }, { x: new Date(2010, 3, 8), y: 85 }
];
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  title='Average Sales Comparison'>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, DateTime]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={dateTimeData} xName='x' yName='y'
name='Sales' type='Line'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Interval Customization

Date time intervals can be customized by using the [interval](#) and [intervalType](#) properties of the axis. For example, when you set interval as 2 and intervalType as years, it considers 2 years as interval.

DateTime axis supports following interval types,

- Auto
- Years
- Months
- Days
- Hours
- Minutes
- Seconds

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";

```

```
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Tooltip, DataLabel, LineSeries } from
'@syncfusion/ej2-react-charts';
import { dateTimeData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'DateTime', title: 'Sales Across
Years', intervalType: 'Years' };
    const primaryyAxis = { title: 'Sales Amount in millions(USD)' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Average Sales Comparison'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, DateTime]}>
            <SeriesCollectionDirective>
                <SeriesDirective dataSource={dateTimeData} xName='x' yName='y'
name='Sales' type='Line'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime', title: 'Sales
Across Years', intervalType: 'Years' };
    const primaryyAxis: AxisModel = { title: 'Sales Amount in millions(USD)'
};
    const dateTimeData: any[] = [
        { x: new Date(2000, 6, 11), y: 10 }, { x: new Date(2002, 3, 7), y: 30 },
        { x: new Date(2004, 3, 6), y: 15 }, { x: new Date(2006, 3, 30), y: 65 },
        { x: new Date(2008, 3, 8), y: 90 }, { x: new Date(2010, 3, 8), y: 85 }
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Average Sales Comparison'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, DateTime]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={dateTimeData} xName='x' yName='y'
name='Sales' type='Line'>
        </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
```

```
ReactDOM.render(<App />, document.getElementById("charts"));
```

Applying Padding to the Range

Padding can be applied to the minimum and maximum extremes of the range by using the [rangePadding](#) property. Date time axis supports the following types of padding,

- None
- Round
- Additional

DateTime - None

When the [rangePadding](#) is set to **None**, minimum and maximum of the axis is based on the data.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Tooltip, DataLabel, LineSeries } from
'@syncfusion/ej2-react-charts';
import { dateTimeData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'DateTime', rangePadding: 'None' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
title='Average Sales Comparison'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, DateTime]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={dateTimeData} xName='x' yName='y'
name='Sales' type='Line'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection }
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime', rangePadding:
'None' };
    const dateTimeData: any[] = [
        { x: new Date(2000, 6, 11), y: 10 }, { x: new Date(2002, 3, 7), y: 30 },
        { x: new Date(2004, 3, 6), y: 15 }, { x: new Date(2006, 3, 30), y: 65 },
```



```

    { x: new Date(2008, 3, 8), y: 90 }, { x: new Date(2010, 3, 8), y: 85 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryXAxis}
    title='Average Sales Comparison'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
    LineSeries, DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={dateTimeData} xName='x' yName='y'
name='Sales' type='Line'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  </>
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));

```

DateTime - Round

When the [rangePadding](#) is set to **Round**, minimum and maximum will be rounded to the nearest possible value divisible by interval.

For example, when the minimum is 15th Jan, interval is 1 and the interval type is 'month', then the axis minimum will be Jan 1st.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Tooltip, DataLabel, LineSeries } from
'@syncfusion/ej2-react-charts';
import { dateTimeData } from 'datasource.ts';
function App() {
  const primaryXAxis = { valueType: 'DateTime', rangePadding: 'Round' };
  return <ChartComponent id='charts' primaryXAxis={primaryXAxis}
title='Average Sales Comparison'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
    LineSeries, DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={dateTimeData} xName='x' yName='y'
name='Sales' type='Line'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";

```

```
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime', rangePadding:
  'Round' };
  const dateTimeData: any[] = [
    { x: new Date(2000, 6, 11), y: 10 }, { x: new Date(2002, 3, 7), y: 30 },
    { x: new Date(2004, 3, 6), y: 15 }, { x: new Date(2006, 3, 30), y: 65 },
    { x: new Date(2008, 3, 8), y: 90 }, { x: new Date(2010, 3, 8), y: 85 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    title='Average Sales Comparison'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={dateTimeData} xName='x' yName='y'
name='Sales' type='Line'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

DateTime - Additional

When the [rangePadding](#) is set to **Additional**, interval of an axis will be padded to the minimum and maximum of the axis.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Tooltip, DataLabel, LineSeries } from
 '@syncfusion/ej2-react-charts';
import { dateTimeData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'DateTime', rangePadding: 'Additional'
};
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
title='Average Sales Comparison'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={dateTimeData} xName='x' yName='y'
name='Sales' type='Line'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
```

```
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime', rangePadding:
'Additional' };
    const dateTimeData: any[] = [
        { x: new Date(2000, 6, 11), y: 10 }, { x: new Date(2002, 3, 7), y: 30 },
        { x: new Date(2004, 3, 6), y: 15 }, { x: new Date(2006, 3, 30), y: 65 },
        { x: new Date(2008, 3, 8), y: 90 }, { x: new Date(2010, 3, 8), y: 85 }
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        title='Average Sales Comparison'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, DateTime]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={dateTimeData} xName='x' yName='y'
name='Sales' type='Line'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Label Format

You can format and parse the date to all globalize format using [Link to the Video](#) property in an axis.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Tooltip, DataLabel, LineSeries } from
'@syncfusion/ej2-react-charts';
import { DateTimeData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'DateTime', labelFormat: 'yMd' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
title='Average Sales Comparison'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, DateTime]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={dateTimeData} xName='x' yName='y'
name='Sales' type='Line'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

```

        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
  AxesDirective, AxisDirective, SeriesDirective, Inject,
  ColumnSeries, Legend, DateTime, Tooltip, DataLabel, Zoom, Crosshair,
  LineSeries, Selection }
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime', labelFormat:
    'yMd' };
  const dateTimeData: any[] = [
    { x: new Date(2000, 6, 11), y: 10 }, { x: new Date(2002, 3, 7), y: 30 },
    { x: new Date(2004, 3, 6), y: 15 }, { x: new Date(2006, 3, 30), y: 65 },
    { x: new Date(2008, 3, 8), y: 90 }, { x: new Date(2010, 3, 8), y: 85 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    title='Average Sales Comparison'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
    LineSeries, DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={dateTimeData} xName='x' yName='y'
name='Sales' type='Line'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  };
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

The following table describes the result of applying some common date time formats to the labelFormat property

<!-- markdownlint-disable MD033 -->

Label Value	Label Format Property Value	Result	Description
new Date(2000, 03, 10)	EEEE	Monday	The Date is displayed in day format
new Date(2000, 03, 10)	yMd	04/10/2000	The Date is displayed in month/date/year format

new Date(2000, 03, 10)	MMM	Apr	The Shorthand month for the date is displayed
new Date(2000, 03, 10)	hm	12:00 AM	Time of the date value is displayed as label
new Date(2000, 03, 10)	hms	12:00:00 AM	The Label is displayed in hours:minutes:seconds format

Logarithmic axis in React Chart component

<!-- markdownlint-disable MD033 -->

Logarithmic axis uses logarithmic scale and it is very useful in visualizing data, when it has numeric values in both lower order of magnitude (eg: 10^{-6}) and higher order of magnitude (eg: 10^6).

To get start quickly with React Logarithmic Axis, you can check out this video:

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject, ColumnSeries, Legend, DateTime, Logarithmic, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-react-charts';
import { logData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'DateTime', title: 'Years', labelFormat: 'y' };
    const primaryyAxis = { valueType: 'Logarithmic', title: 'Profit' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis} primaryYAxis={primaryyAxis} title='Product X Growth [1995-2005]'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Logarithmic, LineSeries, DateTime]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={logData} xName='x' yName='y' name='Product X' type='Line'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective, AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend, DateTime, Logarithmic, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection } from '@syncfusion/ej2-react-charts';
```

```
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime', title: 'Years',
labelFormat: 'y' };
  const primaryyAxis: AxisModel = { valueType: 'Logarithmic', title:
'Profit' };
  const logData: any[] = [
    { x: new Date(1995, 0, 1), y: 80 }, { x: new Date(1996, 0, 1), y: 200 },
    { x: new Date(1997, 0, 1), y: 400 }, { x: new Date(1998, 0, 1), y: 600
  },
    { x: new Date(1999, 0, 1), y: 700 }, { x: new Date(2000, 0, 1), y: 1400
  },
    { x: new Date(2001, 0, 1), y: 2000 }, { x: new Date(2002, 0, 1), y: 4000
  },
    { x: new Date(2003, 0, 1), y: 6000 }, { x: new Date(2004, 0, 1), y: 8000
  },
    { x: new Date(2005, 0, 1), y: 11000 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Product X Growth [1995-2005]'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Logarithmic, LineSeries, DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={logData} xName='x' yName='y'
name='Product X' type='Line'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Note: To use log axis, we need to inject **Logarithmic** module into the **services** and set the **valueType** of axis to **Logarithmic**.

Range

Range of an axis, will be calculated automatically based on the provided data, you can also customize the range of the axis using **minimum**, **maximum** and **interval** property of the axis.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Logarithmic, Tooltip, DataLabel, LineSeries
} from '@syncfusion/ej2-react-charts';
import { logData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'DateTime', title: 'Years',
labelFormat: 'y' };
  const primaryyAxis = { valueType: 'Logarithmic', title: 'Profit',
minimum: 100, maximum: 10000 };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Product X Growth [1995-2005]'>
```

```

    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Logarithmic, LineSeries, DateTime]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={logData} xName='x' yName='y'
name='Product X' type='Line'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Logarithmic, Tooltip, DataLabel, Zoom,
Crosshair, LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime', title: 'Years',
labelFormat: 'y' };
  const primaryyAxis: AxisModel = { valueType: 'Logarithmic', title:
'Profit', minimum: 100, maximum: 10000 };
  const logData: any[] = [
    { x: new Date(1995, 0, 1), y: 80 }, { x: new Date(1996, 0, 1), y: 200 },
    { x: new Date(1997, 0, 1), y: 400 }, { x: new Date(1998, 0, 1), y: 600
  },
    { x: new Date(1999, 0, 1), y: 700 }, { x: new Date(2000, 0, 1), y: 1400
  },
    { x: new Date(2001, 0, 1), y: 2000 }, { x: new Date(2002, 0, 1), y: 4000
  },
    { x: new Date(2003, 0, 1), y: 6000 }, { x: new Date(2004, 0, 1), y: 8000
  },
    { x: new Date(2005, 0, 1), y: 11000 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Product X Growth [1995-2005]'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Logarithmic, LineSeries, DateTime]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={logData} xName='x' yName='y'
name='Product X' type='Line'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  };
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Logarithmic Base

Logarithmic base can be customized by using the [logBase](#) property of the axis. For example when the logBase is 5, the axis values follows 5^{-2} , 5^{-1} , 5^0 , 5^1 , 5^2 etc.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Logarithmic, Tooltip, DataLabel, LineSeries
} from '@syncfusion/ej2-react-charts';
import { logData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'DateTime', title: 'Years',
labelFormat: 'y' };
    const primaryyAxis = { valueType: 'Logarithmic', title: 'Profit',
minimum: 100, maximum: 10000, logBase: 2 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Product X Growth [1995-2005] '>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Logarithmic, LineSeries, DateTime]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={logData} xName='x' yName='y'
name='Product X' type='Line'>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Logarithmic, Tooltip, DataLabel, Zoom,
Crosshair, LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime', title: 'Years',
labelFormat: 'y' };
    const primaryyAxis: AxisModel = { valueType: 'Logarithmic', title:
'Profit', minimum: 100, maximum: 10000, logBase: 2 };
    const logData: any[] = [
        { x: new Date(1995, 0, 1), y: 80 }, { x: new Date(1996, 0, 1), y: 200 },
        { x: new Date(1997, 0, 1), y: 400 }, { x: new Date(1998, 0, 1), y: 600
    },
        { x: new Date(1999, 0, 1), y: 700 }, { x: new Date(2000, 0, 1), y: 1400
    },
        { x: new Date(2001, 0, 1), y: 2000 }, { x: new Date(2002, 0, 1), y: 4000
    },
    ],
    <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Product X Growth [1995-2005] '>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Zoom,
Crosshair, LineSeries, Selection]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={logData} xName='x' yName='y'
name='Product X' type='Line'>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```



```

    { x: new Date(2003, 0, 1), y: 6000 }, { x: new Date(2004, 0, 1), y: 8000
  },
  { x: new Date(2005, 0, 1), y: 11000 }
];
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  title='Product X Growth [1995-2005] '>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Logarithmic, LineSeries, DateTime]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={logData} xName='x' yName='y'
name='Product X' type='Line'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Logarithmic Interval

Logarithmic axis interval can be customized by using the [Link to the Video](#) property of the axis. When the logarithmic base is 10 and logarithmic interval is 2, then the axis labels are placed at an interval of 10^2 . The default value of the interval is 1.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Logarithmic, Tooltip, DataLabel, LineSeries
} from '@syncfusion/ej2-react-charts';
import { logData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'DateTime', title: 'Years',
labelFormat: 'y' };
  const primaryyAxis = { valueType: 'Logarithmic', title: 'Profit',
minimum: 100, maximum: 10000, interval: 2 };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Product X Growth [1995-2005] '>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Logarithmic, LineSeries, DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={logData} xName='x' yName='y'
name='Product X' type='Line'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, DateTime, Logarithmic, Tooltip, DataLabel, Zoom,
Crosshair, LineSeries, Selection }
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime', title: 'Years',
labelFormat: 'y' };
    const primaryyAxis: AxisModel = { valueType: 'Logarithmic', title:
'Profit', minimum: 100, maximum: 10000, interval: 2 };
    const logData: any[] = [
        { x: new Date(1995, 0, 1), y: 80 }, { x: new Date(1996, 0, 1), y: 200 },
        { x: new Date(1997, 0, 1), y: 400 }, { x: new Date(1998, 0, 1), y: 600
    },
        { x: new Date(1999, 0, 1), y: 700 }, { x: new Date(2000, 0, 1), y: 1400
    },
        { x: new Date(2001, 0, 1), y: 2000 }, { x: new Date(2002, 0, 1), y: 4000
    },
        { x: new Date(2003, 0, 1), y: 6000 }, { x: new Date(2004, 0, 1), y: 8000
    },
        { x: new Date(2005, 0, 1), y: 11000 }
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Product X Growth [1995-2005] '>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Logarithmic, LineSeries, DateTime]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={logData} xName='x' yName='y'
name='Product X' type='Line'>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>
    };
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));
}

```

Axis labels in React Chart component

To get start quickly with Axis Labels in React Charts, you can check on this video:

Smart Axis Labels

When the axis labels overlap with each other, you can use [labelIntersectAction](#) property in the axis, to place them smartly.

When setting `labelIntersectAction` as `Hide`

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel } from '@syncfusion/ej2-
react-charts';

```

```
import { smartAxisData } from 'datasource.ts';
function App() {
  const primaryxAxis = { title: 'Countries', labelIntersectAction: 'Hide',
    valueType: 'Category' };
  const primaryyAxis = { title: 'People(in millions)', minimum: 0,
    maximum: 80, interval: 10 };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis} title='Internet Users'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
    Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={smartAxisData} xName='x' yName='y'
        name='Internet' type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
  AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
  Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection } from
  '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { title: 'Countries',
    labelIntersectAction: 'Hide', valueType: 'Category' };
  const primaryyAxis: AxisModel = { title: 'People(in millions)', minimum:
    0, maximum: 80, interval: 10 };
  const smartAxisData: any[] = [
    { x: "South Korea", y: 39.4 }, { x: "India", y: 61.3 }, { x: "Pakistan",
    y: 20.4 },
    { x: "Germany", y: 65.1 }, { x: "Australia", y: 15.8 }, { x: "Italy", y:
    29.2 },
    { x: "France", y: 44.6 }, { x: "Saudi Arabia", y: 9.7 }, { x: "Russia",
    y: 40.8 },
    { x: "Mexico", y: 31 }, { x: "Brazil", y: 75.9 }, { x: "China", y: 51.4
    }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Internet Users'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
    Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={smartAxisData} xName='x' yName='y'
        name='Internet' type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
```

```
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

When setting `labelIntersectAction` as `Rotate45`

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel } from '@syncfusion/ej2-
react-charts';
import { smartAxisData } from 'datasource.ts';
function App() {
    const primaryxAxis = { title: 'Countries', labelIntersectAction:
'Rotate45', valueType: 'Category' };
    const primaryyAxis = { title: 'People(in millions)', minimum: 0,
maximum: 80, interval: 10 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Internet Users'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={smartAxisData} xName='x' yName='y'
name='Internet' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection
} from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { title: 'Countries',
labelIntersectAction: 'Rotate45', valueType: 'Category' };
    const primaryyAxis: AxisModel = { title: 'People(in millions)', minimum:
0, maximum: 80, interval: 10 };
    const smartAxisData: any[] = [
        { x: "South Korea", y: 39.4 }, { x: "India", y: 61.3 }, { x: "Pakistan",
y: 20.4 },
        { x: "Germany", y: 65.1 }, { x: "Australia", y: 15.8 }, { x: "Italy", y:
29.2 },
        { x: "France", y: 44.6 }, { x: "Saudi Arabia", y: 9.7 }, { x: "Russia",
y: 40.8 },
        { x: "Mexico", y: 31 }, { x: "Brazil", y: 75.9 }, { x: "China", y: 51.4
}
    ]
}
```

```

];
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  title='Internet Users'>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={smartAxisData} xName='x' yName='y'
name='Internet' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
);
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

When setting `labelIntersectAction` as `Rotate90`

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel } from '@syncfusion/ej2-
react-charts';
import { smartAxisData } from 'datasource.ts';
function App() {
  const primaryxAxis = { title: 'Countries', labelIntersectAction:
'Rotate90', valueType: 'Category' };
  const primaryyAxis = { title: 'People(in millions)', minimum: 0,
maximum: 80, interval: 10 };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Internet Users'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={smartAxisData} xName='x' yName='y'
name='Internet' type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries,
Selection} from '@syncfusion/ej2-react-charts';
function App() {

```

```

const primaryxAxis: AxisModel = { title: 'Countries',
labelIntersectAction: 'Rotate90', valueType: 'Category' };
const primaryyAxis: AxisModel = { title: 'People(in millions)', minimum:
0, maximum: 80, interval: 10 };
const smartAxisData: any[] = [
  { x: "South Korea", y: 39.4 }, { x: "India", y: 61.3 }, { x: "Pakistan",
y: 20.4 },
  { x: "Germany", y: 65.1 }, { x: "Australia", y: 15.8 }, { x: "Italy", y:
29.2 },
  { x: "France", y: 44.6 }, { x: "Saudi Arabia", y: 9.7 }, { x: "Russia",
y: 40.8 },
  { x: "Mexico", y: 31 }, { x: "Brazil", y: 75.9 }, { x: "China", y: 51.4
}
];
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  title='Internet Users'>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={smartAxisData} xName='x' yName='y'
name='Internet' type='Column'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Axis Labels Positioning

By default, the axis labels can be placed at **outside** the axis line and this also can be placed at **inside** the axis line using the **labelPosition** property.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel } from '@syncfusion/ej2-
react-charts';
import { categoryData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'Category', labelPosition: 'Inside' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;

```

```
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries,
Selection} from '@syncfusion/ej2-react-charts';
function App() {
  const primaryXAxis: AxisModel = { valueType: 'Category', labelPosition:
'Inside' };
  const categoryData: Object[] = [
    { country: "Russia", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "United States", gold: 30, silver: 25, bronze: 27 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryXAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Multilevel Labels

Any number of levels of labels can be added to an axis using the `multiLevelLabels` property.

This property can be configured using the following properties:

- Categories
- Overflow
- Alignment
- Text style
- Border

Note: To use multilevel label feature, we need to inject `MultiLevelLabel` module into the `services`.

Categories

Using the categories property, you can configure the `start`, `end`, `text`, and `maximumTextWidth` of multilevel labels.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, MultiLevelLabel } from
"@syncfusion/ej2-react-charts";
import { categoryData } from "datasource.ts";
function App() {
    const primaryxAxis = {
        valueType: 'Category',
        multiLevelLabels: [{
            categories: [
                {
                    start: -0.5,
                    end: 3.5,
                    //Multi-level label's text.
                    text: 'Half Yearly 1'
                },
                { start: 3.5, end: 7.5, text: 'Half Yearly 2' },
            ]
        }]
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
MultiLevelLabel]}>
            <SeriesCollectionDirective>
                <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
    }
    ;
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection,
MultiLevelLabel } from "@syncfusion/ej2-react-charts";
function App() {
    const primaryxAxis: AxisModel = {
        valueType: 'Category',
        multiLevelLabels: [{
            categories: [
                //Start and end values of the multi-level labels accepts number,
date and string values
            ]
        }]
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
MultiLevelLabel]}>
            <SeriesCollectionDirective>
                <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
    }
    ;
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));
```



```

        start: -0.5,
        end: 3.5,
        //Multi-level label's text.
        text: 'Half Yearly 1'
    },
    { start: 3.5, end: 7.5, text: 'Half Yearly 2' },
]
    ]]
};
const categoryData: Object[] = [
    { country: "Russia", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "United States", gold: 30, silver: 25, bronze: 27 }
];
return <ChartComponent id='charts'
    primaryXAxis={primaryXAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
MultiLevelLabel]} />
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
        </SeriesDirective>
    </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Overflow

Using the `overflow` property, you can `trim` or `wrap` the multilevel labels.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, MultiLevelLabel } from
'@syncfusion/ej2-react-charts';
import { categoryData } from 'datasource.ts';
function App() {
    const primaryXAxis = {
        valueType: 'Category',
        multiLevelLabels: [{
            categories: [{ start: -0.5, end: 3.5, text: 'Half Yearly 1',
maximumTextWidth: 50 },
                { start: 3.5, end: 7.5, text: 'Half Yearly 2',
maximumTextWidth: 50 }],
            overflow: 'Trim'
        }]
    };
    return <ChartComponent id='charts' primaryXAxis={primaryXAxis}>

```

```

    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
MultiLevelLabel]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection,
MultiLevelLabel } from '@syncfusion/ej2-react-charts';
function App() {
  const primaryXAxis: AxisModel = {
    valueType: 'Category',
    multiLevelLabels: [{
      categories: [{ start: -0.5, end: 3.5, text: 'Half Yearly 1',
maximumTextWidth: 50 },
      { start: 3.5, end: 7.5, text: 'Half Yearly 2', maximumTextWidth: 50
}],
      overflow: 'Trim'
    }]
  };
  const categoryData: Object[] = [
    { country: "Russia", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "United States", gold: 30, silver: 25, bronze: 27 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryXAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
MultiLevelLabel]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));

```

Alignment

The **alignment** property provides option to position the multilevel labels at **far**, **center**, or **near**.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, MultiLevelLabel } from
'@syncfusion/ej2-react-charts';
import { categoryData } from 'datasource.ts';
function App() {
    const primaryxAxis = {
        valueType: 'Category',
        multiLevelLabels: [{
            categories: [{ start: -0.5, end: 3.5, text: 'Half Yearly 1'
            },
            { start: 3.5, end: 7.5, text: 'Half Yearly 2' }],
            alignment: 'Far'
        }]
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
MultiLevelLabel]}>/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection,
MultiLevelLabel } from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = {
        valueType: 'Category',
        multiLevelLabels: [{
            categories: [{ start: -0.5, end: 3.5, text: 'Half Yearly 1' },
            { start: 3.5, end: 7.5, text: 'Half Yearly 2' }],
            alignment: 'Far'
        }]
    };
    const categoryData: Object[] = [
        { country: "Russia", gold: 50, silver: 70, bronze: 45 },
        { country: "China", gold: 40, silver: 60, bronze: 55 },
        { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    ];
```

```

    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "United States", gold: 30, silver: 25, bronze: 27 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
MultiLevelLabel]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  };
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Text customization

The `textStyle` property of multilevel labels provides options to customize the size, color, `fontFamily`, `fontWeight`, `fontStyle`, `opacity`, `textAlignment` and `textOverflow`.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, MultiLevelLabel } from
"@syncfusion/ej2-react-charts";
import { categoryData } from 'datasource.ts';
function App() {
  const primaryxAxis = {
    valueType: 'Category',
    multiLevelLabels: [{
      categories: [{ start: -0.5, end: 3.5, text: 'Half Yearly 1'
},
        { start: 3.5, end: 7.5, text: 'Half Yearly 2' } ]],
      textStyle: { size: '18px', color: 'Red' }
    }]
  };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
MultiLevelLabel]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection,
MultiLevelLabel } from '@syncfusion/ej2-react-charts';
function App() {
  const primaryXAxis: AxisModel = {
    valueType: 'Category',
    multiLevelLabels: [{
      categories: [{ start: -0.5, end: 3.5, text: 'Half Yearly 1' },
        { start: 3.5, end: 7.5, text: 'Half Yearly 2' }],
      textStyle: { size: '18px', color: 'Red' }
    }]
  };
  const categoryData: Object[] = [
    { country: "Russia", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "United States", gold: 30, silver: 25, bronze: 27 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryXAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
MultiLevelLabel]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Border customization

Using the **border** property, you can customize the **width**, **color**, and **type**. The type of border are **Rectangle**, **Brace**, **WithoutBorder**, **WithoutTopBorder**, **WithoutTopandBottomBorder** and **CurlyBrace**.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, MultiLevelLabel } from
 '@syncfusion/ej2-react-charts';
import { categoryData } from 'datasource.ts';
function App() {
  const primaryXAxis = {
```

```

        valueType: 'Category',
        multiLevelLabels: [{
            categories: [{ start: -0.5, end: 3.5, text: 'Half Yearly 1'
        },
            { start: 3.5, end: 7.5, text: 'Half Yearly 2' }]],
        border: { type: 'Brace', color: 'Blue', width: 2 },
    }],
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
MultiLevelLabel]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection,
MultiLevelLabel } from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = {
        valueType: 'Category',
        multiLevelLabels: [{
            categories: [{ start: -0.5, end: 3.5, text: 'Half Yearly 1' },
            { start: 3.5, end: 7.5, text: 'Half Yearly 2' }]],
            border: { type: 'Brace', color: 'Blue', width: 2 },
        }],
    };
    const categoryData: Object[] = [
        { country: "Russia", gold: 50, silver: 70, bronze: 45 },
        { country: "China", gold: 40, silver: 60, bronze: 55 },
        { country: "Japan", gold: 70, silver: 60, bronze: 50 },
        { country: "Australia", gold: 60, silver: 56, bronze: 40 },
        { country: "France", gold: 50, silver: 45, bronze: 35 },
        { country: "Germany", gold: 40, silver: 30, bronze: 22 },
        { country: "Italy", gold: 40, silver: 35, bronze: 37 },
        { country: "United States", gold: 30, silver: 25, bronze: 27 }
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
MultiLevelLabel]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>

```

```

        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));

```

Sorting

The chart's data source can be sorted using the `sort` method of chart. The arguments that are required to pass to sort method are data of chart. The fields depend on which sorting is performed either `x` or `y`, and the `isDescending` with which data source values are sorted in either `ascending` or `descending`.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel } from '@syncfusion/ej2-
react-charts';
import { categoryData } from 'datasource.ts';
function App() {
  const primaryXAxis = { valueType: 'Category' };
  return <ChartComponent id='charts' primaryXAxis={primaryXAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection
} from '@syncfusion/ej2-react-charts';
function App() {
  const primaryXAxis: AxisModel = { valueType: 'Category' };
  const categoryData: Object[] = [
    { country: "Russia", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "United States", gold: 30, silver: 25, bronze: 27 }
  ];

```

```

];
return <ChartComponent id='charts'
  primaryXAxis={primaryXAxis}>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Edge Label Placement

Labels with long text at the edges of an axis may appear partially in the chart. To avoid this, use [edgeLabelPlacement](#) property in axis, which moves the label inside the chart area for better appearance or hides it.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel } from '@syncfusion/ej2-
react-charts';
import { categoryData } from 'datasource.ts';
function App() {
  const primaryXAxis = { valueType: 'Category', edgeLabelPlacement:
'Shift' };
  return <ChartComponent id='charts' primaryXAxis={primaryXAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries,
Selection} from '@syncfusion/ej2-react-charts';
function App() {

```



```

const primaryxAxis: AxisModel = { valueType: 'Category',
edgeLabelPlacement: 'Shift' };
const categoryData: Object[] = [
  { country: "Russia", gold: 50, silver: 70, bronze: 45 },
  { country: "China", gold: 40, silver: 60, bronze: 55 },
  { country: "Japan", gold: 70, silver: 60, bronze: 50 },
  { country: "Australia", gold: 60, silver: 56, bronze: 40 },
  { country: "France", gold: 50, silver: 45, bronze: 35 },
  { country: "Germany", gold: 40, silver: 30, bronze: 22 },
  { country: "Italy", gold: 40, silver: 35, bronze: 37 },
  { country: "United States", gold: 30, silver: 25, bronze: 27 }
];
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Labels Customization

Border of the axis labels can be customized using **width**, **color** and **type** property of the axis.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel } from '@syncfusion/ej2-
react-charts';
import { categoryData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'Category', border: { width: 1, type:
'Rectangle', color: 'red' } };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries,
Selection} from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'Category', border: { width:
1, type: 'Rectangle', color: 'red' } };
  const categoryData: Object[] = [
    { country: "Russia", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "United States", gold: 30, silver: 25, bronze: 27 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  </>;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Customizing Specific Point

You can customize the specific text in the axis labels using `axisLabelRender` event.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel } from '@syncfusion/ej2-
react-charts';
import { categoryData } from 'datasource.ts';
function App() {
  const axisLabelRender = (args) => {
    if (args.text === 'France') {
      args.labelStyle.color = 'Red';
    }
  };
  const primaryxAxis = { valueType: 'Category', border: { width: 1, type:
'Rectangle', color: 'red' } };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
axisLabelRender={axisLabelRender}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>

```

```

        <SeriesCollectionDirective>
            <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
IAxisLabelRenderEventArgs, ColumnSeries, Legend, Category, Tooltip,
DataLabel, Zoom, Crosshair, LineSeries, Selection } from '@syncfusion/ej2-
react-charts';
function App() {
    const axisLabelRender = (args: IAxisLabelRenderEventArgs) => {
        if (args.text === 'France') {
            args.labelStyle.color = 'Red';
        }
    }
    const primaryXAxis: AxisModel = { valueType: 'Category', border: { width:
1, type: 'Rectangle', color: 'red' } };
    const categoryData: Object[] = [
        { country: "Russia", gold: 50, silver: 70, bronze: 45 },
        { country: "China", gold: 40, silver: 60, bronze: 55 },
        { country: "Japan", gold: 70, silver: 60, bronze: 50 },
        { country: "Australia", gold: 60, silver: 56, bronze: 40 },
        { country: "France", gold: 50, silver: 45, bronze: 35 },
        { country: "Germany", gold: 40, silver: 30, bronze: 22 },
        { country: "Italy", gold: 40, silver: 35, bronze: 37 },
        { country: "United States", gold: 30, silver: 25, bronze: 27 }
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryXAxis}
        axisLabelRender={axisLabelRender}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Trim using maximum label width

You can trim the label using [enableTrim](#) property and width of the labels can also be customized using [maximumLabelWidth](#) property in the axis, the value maximum label width is **34** by default.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel } from '@syncfusion/ej2-
react-charts';
import { smartAxisData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Category',
        //label is trimmed
        enableTrim: true,
        //maximum width of label is provided
        maximumLabelWidth: '34',
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={smartAxisData} xName='x' yName='y'
type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
IAxisLabelRenderEventArgs, ColumnSeries, Legend, Category, Tooltip,
DataLabel, Zoom, Crosshair, LineSeries, Selection } from '@syncfusion/ej2-
react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Category',
        //label is trimmed
        enableTrim: true,
        //maximum width of label is provided
        maximumLabelWidth: '34',
    };
    const smartAxisData: any[] = [
        { x: "South Korea", y: 39.4 }, { x: "India", y: 61.3 }, { x: "Pakistan",
y: 20.4 },
        { x: "Germany", y: 65.1 }, { x: "Australia", y: 15.8 }, { x: "Italy", y:
29.2 },
        { x: "France", y: 44.6 }, { x: "Saudi Arabia", y: 9.7 }, { x: "Russia",
y: 40.8 },
    ];
```

```

    { x: "Mexico", y: 31 }, { x: "Brazil", y: 75.9 }, { x: "China", y: 51.4
  }
];
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={smartAxisData} xName='x' yName='y'
type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
</>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Line break support

Line break feature used to customize the long axis label text into multiple lines by using tag. Refer the below example in that dataSource x value contains long text, it breaks into two lines by using `
` tag.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
StackingColumnSeries, ColumnSeries, Legend, Category, Tooltip, DataLabel }
from '@syncfusion/ej2-react-charts';
function App() {
  const data1 = [
    { x: 'Egg', y: 106 },
    { x: 'Fish', y: 103 },
    { x: 'Misc', y: 198 },
    { x: 'Tea', y: 189 },
    { x: 'Fruits', y: 250 }
  ];
  const primaryXAxis = { valueType: 'Category', interval: 1, tickPosition:
'Inside', labelPosition: 'Inside', labelStyle: { color: 'ffffff' } };
  const primaryYAxis = { minimum: 0, maximum: 300, interval: 50,
labelStyle: { color: 'transparent' } };
  let count = 0;
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Trade in Food Groups'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
StackingColumnSeries]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data1}
type='Column' xName='x' width={2} yName='y' name='Tiger' cornerRadius={{
bottomLeft: 10, bottomRight: 10, topLeft: 10, topRight: 10 }} marker={{
dataLabel: { visible: true, position: 'Top', font: { fontWeight: '600',
color: 'ffffff' } } }}>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}

```

```
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% enddraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
IAxisLabelRenderEventArgs, StackingColumnSeries, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
  const data1: any[] = [
    { x: 'Egg', y: 106 },
    { x: 'Fish', y: 103 },
    { x: 'Misc', y: 198 },
    { x: 'Tea', y: 189 },
    { x: 'Fruits', y: 250 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category', interval:
1, tickPosition: 'Inside', labelPosition: 'Inside', labelStyle: { color:
'#ffffff' } };
  const primaryyAxis: AxisModel = { minimum: 0, maximum: 300, interval: 50,
labelStyle: { color: 'transparent' } };
  let count: number = 0;

  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis} primaryYAxis={primaryyAxis} title='Trade
in Food Groups'
  >
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category, StackingColumnSeries]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data1}
type='Column' xName='x' width={2} yName='y' name='Tiger'
cornerRadius={{ bottomLeft: 10, bottomRight:
10, topLeft: 10, topRight: 10 }}
marker={{ dataLabel: { visible: true,
position: 'Top', font: { fontWeight: '600', color: '#ffffff' } } }}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% enddraw %}
```

Axis customization in React Chart component

Axis Crossing

An axis can be positioned in the chart area using [crossesAt](#) and [crossesInAxis](#) properties. The [crossesAt](#) property specifies the values (datetime, numeric, or logarithmic) at which the axis line has to be intersected with the vertical axis or vice-versa, and the [crossesInAxis](#) property specifies the axis name with which the axis line has to be crossed.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel } from '@syncfusion/ej2-
react-charts';
import { stripData } from 'datasource.ts';
function App() {
    const primaryxAxis = { crossesAt: 15 };
    const primaryyAxis = { crossesAt: 5 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={stripData} xName='x' yName='y'
type='Column'>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
    }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection } from
'@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { crossesAt: 15 };
    const primaryyAxis: AxisModel = { crossesAt: 5 };
    const stripData: any[] = [
        {x: 1, y: 20},
        {x: 2, y: 22},
        {x: 3, y: 0},
        {x: 4, y: 12},
        {x: 5, y: 5},
        {x: 6, y: 15},
        {x: 7, y: 6},
        {x: 8, y: 12},
        {x: 9, y: 34},
        {x: 10, y: 7}
    ]
}
```

```

];
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={stripData} xName='x' yName='y'
type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
</return>
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Title

You can add a title to the axis using [title](#) property to provide quick information to the user about the data plotted in the axis. Title style can be customized using [titleStyle](#) property of the axis.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel } from '@syncfusion/ej2-
react-charts';
import { categoryData } from 'datasource.ts';
function App() {
  const primaryxAxis = {
    valueType: 'Category', title: 'Countries', titleStyle: {
      size: '16px', color: 'blue', fontFamily: 'Segoe UI', fontWeight:
'bold'
    }
  };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,

```



```

Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection } from
'@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = {
    valueType: 'Category', title: 'Countries', titleStyle: {
      size: '16px', color: 'blue', fontFamily: 'Segoe UI', fontWeight:
'bold'
    }
  };
  const categoryData: Object[] = [
    { country: "Russia", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "United States", gold: 30, silver: 25, bronze: 27 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Title Rotation

By using the [titleRotation](#) property, you can rotate the axis title from 0 to 360 degree.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel } from '@syncfusion/ej2-
react-charts';
import { categoryData } from 'datasource.ts';
class App extends React.Component {
  primaryxAxis = {
    valueType: 'Category', title: 'Countries', titleRotation: 90,
titleStyle: {
      size: '16px', color: 'blue', fontFamily: 'Segoe UI', fontWeight:
'bold'
    }
  };
  render() {
    return <ChartComponent id='charts' primaryXAxis={this.primaryxAxis}>
      <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    </ChartComponent>
  };
}

```

```

        <SeriesCollectionDirective>
            <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
}
;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection } from
'@syncfusion/ej2-react-charts';
class App extends React.Component<{}, {}> {
    public primaryxAxis: AxisModel = {
        valueType: 'Category', title: 'Countries', titleRotation: 90,
        titleStyle: {
            size: '16px', color: 'blue', fontFamily: 'Segoe UI', fontWeight:
'bold'
        }
    };
    public categoryData: any[] = [{ country: "Russia", gold: 50, silver: 70,
bronze: 45 }, { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 }, { country:
"Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 }, { country:
"Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 }, { country:
"United States", gold: 30, silver: 25, bronze: 27 }];
    render() {
        return <ChartComponent id='charts'
            primaryXAxis={this.primaryxAxis}>
            <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
            <SeriesCollectionDirective>
                <SeriesDirective dataSource={this.categoryData} xName='country'
yName='gold' type='Column'>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>
    }
};
ReactDOM.render(<App />, document.getElementById("charts"));

```

Tick Lines Customization

You can customize the [width](#), and [color](#) of the minor and major tick lines, using [majorTickLines](#) and [minorTickLines](#) properties in the axis.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel } from '@syncfusion/ej2-
react-charts';
import { categoryData } from 'datasource.ts';
function App() {
    const primaryxAxis = {
        valueType: 'Category',
        majorTickLines: {
            color: 'blue',
            width: 5
        },
        minorTickLines: {
            color: 'red',
            width: 0
        }
    };
    const primaryyAxis = {
        title: 'Temperature (Fahrenheit)',
        majorTickLines: {
            color: 'blue',
            width: 5
        },
        minorTickLines: {
            color: 'red',
            width: 0
        }
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Temperature flow over months'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' name='Sales' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection } from
 '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = {
        valueType: 'Category',
        majorTickLines: {

```

```

        color: 'blue',
        width: 5
    },
    minorTickLines: {
        color: 'red',
        width: 0
    }
};
const primaryyAxis: AxisModel = {
    title: 'Temperature (Fahrenheit)',
    majorTickLines: {
        color: 'blue',
        width: 5
    },
    minorTickLines: {
        color: 'red',
        width: 0
    }
};
const categoryData: Object[] = [
    { country: "Russia", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "United States", gold: 30, silver: 25, bronze: 27 }
];
return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Temperature flow over months'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' name='Sales' type='Column'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Grid Lines Customization

You can customize the [width](#), [color](#) and [dashArray](#) of the minor and major grid lines using [majorGridLines](#) and [minorGridLines](#) properties in the axis.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel } from '@syncfusion/ej2-
react-charts';

```

```

import { categoryData } from 'datasource.ts';
function App() {
  const primaryxAxis = {
    valueType: 'Category',
    majorGridLines: {
      color: 'blue',
      width: 1
    },
    minorGridLines: {
      color: 'red',
      width: 0
    }
  };
  const primaryyAxis = {
    title: 'Temperature (Fahrenheit)',
    majorGridLines: {
      color: 'blue',
      width: 1
    },
    minorGridLines: {
      color: 'red',
      width: 0
    }
  };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis} title='Temperature flow over months'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
    Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={categoryData} xName='country'
      yName='gold' name='Sales' type='Column'>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection } from
'@syncfusion/ej2-react-charts';
import { categoryData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = {
    valueType: 'Category',
    majorGridLines: {
      color: 'blue',
      width: 1
    },
    minorGridLines: {

```

```

        color: 'red',
        width: 0
    }
};
const primaryyAxis: AxisModel = {
    title: 'Temperature (Fahrenheit)',
    majorGridLines: {
        color: 'blue',
        width: 1
    },
    minorGridLines: {
        color: 'red',
        width: 0
    }
};
const categoryData: any[] = [
    { country: "Russia", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "United States", gold: 30, silver: 25, bronze: 27 }
];
return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Temperature flow over months'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={categoryData} xName='country'
yName='gold' name='Sales' type='Column'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Multiple Axis

In addition to primary X and Y axis, we can add n number of axis to the chart. Series can be associated with this axis, by mapping with axis's unique name.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend, Category,
Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-react-charts';
import { PaneData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Category' };
    const primaryyAxis = {

```

```

        title: 'Temperature (Fahrenheit)', minimum: 0, maximum: 90,
        interval: 10,
        lineStyle: { width: 0 }, labelFormat: '{value}°F'
    };
    const marker = { visible: true, width: 10, height: 10, border: { width:
2, color: '#F8AB1D' } };
    const lines = { width: 0 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Temperature flow over months'>
    <Inject services={[ColumnSeries, LineSeries, Legend, Tooltip,
DataLabel, Category]}/>
    <AxesDirective>
    <AxisDirective rowIndex={0} name='yAxis1' opposedPosition={true}
title='Temperature (Celsius)' majorGridLines={lines} lineStyle={lines}>
    </AxisDirective>
    </AxesDirective>
    <SeriesCollectionDirective>
    <SeriesDirective dataSource={PaneData} xName='x' yName='y'
name='Germany' type='Column'>
    </SeriesDirective>
    <SeriesDirective dataSource={PaneData} xName='x' yName='y1'
name='Japan' type='Line' marker={marker} yAxisName='yAxis1'>
    </SeriesDirective>
    </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection } from
'@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Category' };
    const primaryyAxis: AxisModel = {
        title: 'Temperature (Fahrenheit)', minimum: 0, maximum: 90, interval:
10,
        lineStyle: { width: 0 }, labelFormat: '{value}°F'
    };
    const marker = { visible: true, width: 10, height: 10, border: { width: 2,
color: '#F8AB1D' } };
    const lines = { width: 0 };
    const PaneData: any[] = [
        { x: 'Jan', y: 15, y1: 33 }, { x: 'Feb', y: 20, y1: 31 }, { x: 'Mar', y:
35, y1: 30 },
        { x: 'Apr', y: 40, y1: 28 }, { x: 'May', y: 80, y1: 29 }, { x: 'Jun', y:
70, y1: 30 },
        { x: 'Jul', y: 65, y1: 33 }, { x: 'Aug', y: 55, y1: 32 }, { x: 'Sep', y:
50, y1: 34 },

```

```

    { x: 'Oct', y: 30, y1: 32 }, { x: 'Nov', y: 35, y1: 32 }, { x: 'Dec', y:
35, y1: 31 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Temperature flow over months'>
    <Inject services={[ColumnSeries, LineSeries, Legend, Tooltip,
DataLabel, Category]} />
    <AxesDirective>
      <AxisDirective rowIndex={0} name='yAxis1' opposedPosition={true}
title='Temperature (Celsius)'
        majorGridLines={lines} lineStyle={lines} >
      </AxisDirective>
    </AxesDirective>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={PaneData} xName='x' yName='y'
name='Germany' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={PaneData} xName='x' yName='y1'
name='Japan' type='Line'
        marker={marker} yAxisName='yAxis1' >
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Inversed Axis

When an axis is inversed, highest value of the axis comes closer to origin and vice versa. To place an axis in inversed manner set this property `isInversed` to true.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, DataLabel } from '@syncfusion/ej2-react-
charts';
import { inverseData } from 'datasource.ts';
function App() {
    const primaryxAxis = { title: 'Years', opposedPosition: true,
isInversed: true };
    const primaryyAxis = { title: 'Exchange rate (INR per USD)', isInversed:
true };
    const marker = { dataLabel: { visible: true } };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Exchange Rate'>
        <Inject services={[ColumnSeries, Category, Legend, DataLabel]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={inverseData} xName='x' yName='y'
type='Column' marker={marker}>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>
    </App>
}
```



```

    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, DataLabel } from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { title: 'Years', opposedPosition: true,
isInversed: true };
  const primaryyAxis: AxisModel = { title: 'Exchange rate (INR per USD)',
isInversed: true };
  const marker = { dataLabel: { visible: true } };
  const inverseData: any[] = [
    { x: 2008, y: 15.1 }, { x: 2009, y: 16 }, { x: 2010, y: 21.4 },
    { x: 2011, y: 18 }, { x: 2012, y: 16.2 }, { x: 2013, y: 11 },
    { x: 2014, y: 7.6 }, { x: 2015, y: 1.5 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Exchange Rate'>
    <Inject services={[ColumnSeries, Category, Legend, DataLabel]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={inverseData} xName='x' yName='y'
type='Column' marker={marker}>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  </>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Opposed Position

To place an axis opposite from its original position, set [opposedPosition](#) property of the axis to true.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, DataLabel } from '@syncfusion/ej2-react-
charts';
import { inverseData } from 'datasource.ts';
function App() {
  const primaryxAxis = { title: 'Years', opposedPosition: true };
  const primaryyAxis = { title: 'Exchange rate (INR per USD)' };
  const marker = { dataLabel: { visible: true } };

```

```

    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis} title='Exchange Rate'>
      <Inject services={[ColumnSeries, Category, Legend, DataLabel]}/>
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={inverseData} xName='x' yName='y'
type='Column' marker={marker}>
          </SeriesDirective>
        </SeriesCollectionDirective>
      </ChartComponent>;
    }
    ;
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend,
Category, DataLabel } from '@syncfusion/ej2-react-charts';
import { inverseData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { title: 'Years', opposedPosition: true };
  const primaryyAxis: AxisModel = { title: 'Exchange rate (INR per USD)' };
  const marker = { dataLabel: { visible: true } };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Exchange Rate'>
    <Inject services={[ColumnSeries, Category, Legend, DataLabel]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={inverseData} xName='x' yName='y'
type='Column' marker={marker}>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));

```

<!-- markdownlint-disable MD036 -->

Strip line in React Chart component

<!-- markdownlint-disable MD036 -->

EJ2 chart supports horizontal and vertical strip lines and customization of stripline in both orientation.

To use strip line feature, you need to inject **StripLine** into the **services**.

Horizontal Strip lines

You can create Horizontal stripline by adding the **stripline** in the vertical axis and set **visible** option to true. Striplines are rendered in the specified start to end range and you can add more than one stripline for an axis.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
StripLine, ColumnSeries, Legend, Category, Tooltip, DataLabel } from
'@syncfusion/ej2-react-charts';
import { stripData } from 'datasource.ts';
function App() {
    const primaryxAxis = { title: 'Overs' };
    const primaryyAxis = {
        title: 'Runs',
        stripLines: [{ start: 15, end: 22, text: 'Good', color: '#ff512f',
visible: true },
            { start: 8, end: 15, text: 'Medium', color: 'pink', opacity:
0.5, visible: true },
            { start: 0, end: 8, text: 'Not enough', color: 'skyblue',
opacity: 0.5, visible: true } ]
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='India Vs Australia 1st match'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
StripLine]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={stripData} xName='x' yName='y'
type='Column'>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
    }
    ;
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, StripLine,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection, StripLinesDirective, StripLineDirective }
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { title: 'Overs' };
    const primaryyAxis: AxisModel = {
        title: 'Runs',
        stripLines: [{ start: 15, end: 22, text: 'Good', color: '#ff512f',
visible: true },
            { start: 8, end: 15, text: 'Medium', color: 'pink', opacity: 0.5,
visible: true },
            { start: 0, end: 8, text: 'Not enough', color: 'skyblue', opacity: 0.5,
visible: true } ]
    };
    const stripData: any[] = [
        {x: 1, y: 20}, {x: 2, y: 22}, {x: 3, y: 0}, {x: 4, y: 12}, {x: 5, y: 5},
```

```

    {x: 6, y: 15}, {x: 7, y: 6}, {x: 8, y: 12}, {x: 9, y: 34}, {x: 10, y: 7}
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='India Vs Australia 1st match'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
StripLine]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={stripData} xName='x' yName='y'
type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  </>
  export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Vertical Striplines

You can create vertical stripline by adding the `stripline` in the horizontal axis and set `visible` option to `true`. Striplines are rendered in the specified start to end range and you can add more than one stripline for an axis.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
StripLine, ColumnSeries, Legend, Category, Tooltip, DataLabel } from
'@syncfusion/ej2-react-charts';
import { stripData } from 'datasource.ts';
function App() {
  const primaryxAxis = {
    title: 'Overs', stripLines: [{ start: 0, end: 5, text: 'powerplay
1', color: 'red', visible: true, opacity: 0.5 },
    { start: 5, end: 10, text: 'powerplay 2', color: 'blue',
visible: true, opacity: 0.5 }]
  };
  const primaryyAxis = { title: 'Runs' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='India Vs Australia 1st match'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
StripLine]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={stripData} xName='x' yName='y'
type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, StripLine,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection, StripLinesDirective, StripLineDirective }
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = {
    title: 'Overs', stripLines: [{ start: 0, end: 5, text: 'powerplay 1',
color: 'red', visible: true, opacity: 0.5 },
{ start: 5, end: 10, text: 'powerplay 2', color: 'blue', visible: true,
opacity: 0.5 } ]
  };
  const primaryyAxis: AxisModel = { title: 'Runs' };
  const stripData: any[] = [
    {x: 1, y: 20}, {x: 2, y: 22}, {x: 3, y: 0}, {x: 4, y: 12}, {x: 5, y: 5},
    {x: 6, y: 15}, {x: 7, y: 6}, {x: 8, y: 12}, {x: 9, y: 34}, {x: 10, y: 7}
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='India Vs Australia 1st match'>
    <Inject services=[ColumnSeries, Legend, Tooltip, DataLabel, Category,
StripLine] />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={stripData} xName='x' yName='y'
type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  </>
  export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Customize the strip line

Starting value in specific strip line can be customized by **start** property in strip line. Similarly, ending value is customized by **end**. It can be also set for starting from the corresponding origin of the axis by **startFromOrigin**. Size of the strip line is customized by **size**. Border for the stripline is customized by **border**. Order of the strip line such that whether it should be rendered in behind or over the series elements is customized by **zIndex**.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
StripLine, ColumnSeries, Legend, Category, Tooltip, DataLabel } from
"@syncfusion/ej2-react-charts";
import { stripData } from "datasource.ts";
function App() {
  const primaryxAxis = { title: 'Overs', stripLines: [{ startFromOrigin:
true, size: 4, zIndex: 'Behind', opacity: 0.5 } ] };
  const primaryyAxis = { title: 'Runs' };
```

```

    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='India Vs Australia 1st match'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
StripLine]}/>
    <SeriesCollectionDirective>
    <SeriesDirective dataSource={stripData} xName='x' yName='y'
type='Column'>
    </SeriesDirective>
    </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, StripLine,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection, StripLinesDirective, StripLineDirective}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { title: 'Overs', stripLines: [{
startFromOrigin: true, size: 4, zIndex: 'Behind', opacity: 0.5 }] };
    const primaryyAxis: AxisModel = { title: 'Runs' };
    const stripData: any[] = [
        {x: 1, y: 20},{x: 2, y: 22},{x: 3, y: 0},{x: 4, y: 12},{x: 5, y: 5},
        {x: 6, y: 15},{x: 7, y: 6},{x: 8, y: 12},{x: 9, y: 34},{x: 10, y: 7}
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='India Vs Australia 1st match'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
StripLine]}/>
        <SeriesCollectionDirective>
        <SeriesDirective dataSource={stripData} xName='x' yName='y'
type='Column'>
        </SeriesDirective>
        </SeriesCollectionDirective>
        </ChartComponent>
    };
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));
}

```

Customize the stripline text

You can customize the text rendered in stripline by `textStyle` property. Rotation of the strip line text can be changed by `rotation` property. Horizontal and Vertical alignment of stripline text can be changed by `horizontalAlignment` and `verticalAlignment` property.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
StripLine, ColumnSeries, Legend, Category, Tooltip, DataLabel } from
'@syncfusion/ej2-react-charts';
import { stripData } from 'datasource.ts';
function App() {
    const primaryxAxis = {
        title: 'Overs', stripLines: [
            {
                startFromOrigin: true, size: 4, zIndex: 'Behind', opacity:
0.5, verticalAlignment: 'Middle', horizontalAlignment: 'End', rotation: 90,
                textStyle: { size: 15 }
            }, { start: 5, end: 8, verticalAlignment: 'Start',
horizontalAlignment: 'End', rotation: 45 }
        ]
    };
    const primaryyAxis = { title: 'Runs' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='India Vs Australia 1st match'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
StripLine]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={stripData} xName='x' yName='y'
type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxisDirective, AxisDirective, SeriesDirective, Inject, StripLine,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection, StripLinesDirective, StripLineDirective}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = {
        title: 'Overs', stripLines: [
            {
                startFromOrigin: true, size: 4, zIndex: 'Behind', opacity: 0.5,
verticalAlignment: 'Middle', horizontalAlignment: 'End', rotation: 90,
                textStyle: { size: 15 }
            }, { start: 5, end: 8, verticalAlignment: 'Start',
horizontalAlignment: 'End', rotation: 45 }
        ]
    };
    const primaryyAxis: AxisModel = { title: 'Runs' };
    const stripData: any[] = [
        {x: 1, y: 20}, {x: 2, y: 22}, {x: 3, y: 0}, {x: 4, y: 12}, {x: 5, y: 5},
        {x: 6, y: 15}, {x: 7, y: 6}, {x: 8, y: 12}, {x: 9, y: 34}, {x: 10, y: 7}
    ];

```

```

];
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  title='India Vs Australia 1st match'>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
StripLine]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={stripData} xName='x' yName='y'
type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
</return>
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Dash Array

You can create dash array stripline by using `dashArray` property. The Striplines are rendered with specified dash array values.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
StripLine, ColumnSeries, Legend, Category, Tooltip, DataLabel } from
"@syncfusion/ej2-react-charts";
import { stripData } from "datasource.ts";
function App() {
  const primaryyAxis = { minimum: 0, maximum: 60, interval: 10,
    stripLines: [{ start: 30, size: 2, sizeType: 'Pixel', dashArray:
"10,5", color: "red" }]
  };
  const marker = { visible: true };
  return <ChartComponent id='charts' primaryYAxis={primaryyAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
StripLine]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={stripData} xName='x' yName='y'
type='Column' marker={marker}>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, StripLine,

```



```

ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection, StripLinesDirective, StripLineDirective}
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryyAxis: AxisModel = { minimum: 0, maximum: 60, interval: 10,
    stripLines:[{ start: 30, size: 2, sizeType: 'Pixel',
dashArray:"10,5", color: "red"}]
  };
  const marker = { visible: true };
  const stripData: any[] = [
    {x: 1, y: 20},{x: 2, y: 22},{x: 3, y: 0},{x: 4, y: 12},{x: 5, y: 5},
    {x: 6, y: 15},{x: 7, y: 6},{x: 8, y: 12},{x: 9, y: 34},{x: 10, y: 7}
  ];
  return <ChartComponent id='charts'
    primaryYAxis={primaryyAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
StripLine]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={stripData} xName='x' yName='y'
type='Column' marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Recurrence Stripline

The strip lines to be drawn repeatedly at the regular intervals – this will be useful when you want to mark an event that occurs recursively along the timeline of the chart. Following properties are used to configure this feature.

- **isRepeat** - It is used for enable / disable the recurrence strip line.
- **repeatEvery** - It is used for mention the stripline interval.
- **repeatUntil** - It specifies the end value at which point strip line has to stop repeating.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
StripLine, ColumnSeries, Legend, Category, Tooltip, DataLabel, LineSeries }
from '@syncfusion/ej2-react-charts';
function App() {
  const chartData = [{ x: 1, y: 5 }, { x: 2, y: 39 }, { x: 3, y: 21 }, {
x: 4, y: 51 }, { x: 5, y: 30 },
    { x: 6, y: 25 }, { x: 7, y: 10 }, { x: 8, y: 40 }, { x: 9, y: 50 },
    { x: 10, y: 20 }];
  const primaryxAxis = { stripLines: [{ start: 1, size: 1, isRepeat: true,
repeatEvery: 2, color: 'rgba(167,169,171, 0.3)' }]
  };
  const marker = { visible: true };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>

```

```

    <Inject services={[ColumnSeries, LineSeries, Legend, Tooltip,
DataLabel, Category, StripLine]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='x' yName='y'
type='Line' marker={marker}>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, StripLine,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection, StripLinesDirective, StripLineDirective}
from '@syncfusion/ej2-react-charts';
function App() {
  const chartData: any[] = [{x: 1, y: 5},{x: 2, y: 39},{x: 3, y: 21},{x: 4,
y: 51},{x: 5, y: 30},
                                {x: 6, y: 25},{x: 7, y: 10},{x: 8, y: 40},{x:
9, y: 50},{x: 10, y: 20}];
  const primaryXAxis: AxisModel = { stripLines:[{start: 1, size: 1,
isRepeat: true, repeatEvery: 2, color: 'rgba(167,169,171, 0.3)'}]}
  };
  const marker = { visible: true };
  return <ChartComponent id='charts'
    primaryXAxis={primaryXAxis}>
    <Inject services={[ColumnSeries, LineSeries, Legend, Tooltip,
DataLabel, Category, StripLine]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='x' yName='y'
type='Line' marker={marker}>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));

```

Size Type

The `sizeType` property refers the size of the stripline. They are,

- Auto
- Pixel
- Years
- Months
- Days

- Hours
- Minutes
- Seconds

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
StripLine, DateTime, ColumnSeries, Legend, Tooltip, DataLabel, LineSeries }
from '@syncfusion/ej2-react-charts';
function App() {
    const chartData = [{ x: new Date(2000, 0, 1), y: 10 }, { x: new
Date(2002, 0, 1), y: 40 },
        { x: new Date(2004, 0, 1), y: 20 }, { x: new Date(2006, 0, 1), y: 50
    },
        { x: new Date(2008, 0, 1), y: 15 }, { x: new Date(2010, 0, 1), y: 30
    }
    ];
    const primaryxAxis = { valueType: 'DateTime', intervalType: 'Years',
        stripLines: [{ start: new Date(2002, 0, 1), size: 2, sizeType:
'Years', color: 'rgba(167,169,171, 0.3)' }] };
    const primaryyAxis = { minimum: 0, maximum: 60, interval: 10 };
    const marker = { visible: true };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis}>
        <Inject services={[ColumnSeries, Legend, LineSeries, Tooltip,
DataLabel, DateTime, StripLine]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} xName='x' yName='y'
type='Line' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, StripLine, DateTime,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection, StripLinesDirective, StripLineDirective}
from '@syncfusion/ej2-react-charts';
function App() {
    const chartData: any[] = [{ x: new Date(2000, 0, 1), y: 10 }, { x: new
Date(2002, 0, 1), y: 40 },
        { x: new Date(2004, 0, 1), y: 20 }, { x: new Date(2006, 0, 1), y: 50 },
        { x: new Date(2008, 0, 1), y: 15 }, { x: new Date(2010, 0, 1), y: 30
    }
    ];
    const primaryxAxis: AxisModel = { valueType: 'DateTime', intervalType:
'Years',
```

```

        stripLines:[{start:new Date(2002, 0, 1) , size: 2, sizeType: 'Years',
color: 'rgba(167,169,171, 0.3)'}] };
    const primaryyAxis: AxisModel={ minimum: 0, maximum: 60, interval: 10 }
    const marker = { visible: true };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}>
        <Inject services={[ColumnSeries, Legend, LineSeries, Tooltip,
DataLabel, DateTime, StripLine]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} xName='x' yName='y'
type='Line' marker={marker}>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>
    );
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Segment Stripline

You can create stripline in a particular region with respect to segment. You can enable the segment stripline using `isSegmented` property. The start and end value of this type of stripline can be defined using `segmentStart` and `segmentEnd` properties.

- `isSegmented` - It is used for enable the segment stripline.
- `segmentStart` - Used to change the segment start value. Value correspond to associated axis.
- `segmentEnd` - Used to change the segment end value. Value correspond to associated axis.
- `segmentAxisName` - Used to specify the name of the associated axis.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
StripLine, ColumnSeries, Legend, Category, Tooltip, DataLabel, LineSeries }
from '@syncfusion/ej2-react-charts';
function App() {
    const chartData = [{ x: 1, y: 5 }, { x: 2, y: 39 }, { x: 3, y: 21 }, {
x: 4, y: 51 }, { x: 5, y: 30 },
        { x: 6, y: 25 }, { x: 7, y: 10 }, { x: 8, y: 40 }, { x: 9, y: 50 },
{ x: 10, y: 20 }];
    const primaryyAxis = { stripLines: [
        { start: 20, end: 40, isSegmented: true, segmentStart: 2,
segmentEnd: 4,
            color: 'rgba(167,169,171, 0.3)' }
    ]
    };
    const marker = { visible: true };
    return <ChartComponent id='charts' primaryYAxis={primaryyAxis}>
        <Inject services={[ColumnSeries, LineSeries, Legend, Tooltip,
DataLabel, Category, StripLine]} />
        <SeriesCollectionDirective>

```

```

        <SeriesDirective dataSource={chartData} xName='x' yName='y'
type='Line' marker={marker}>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject, StripLine,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection, StripLinesDirective, StripLineDirective }
from '@syncfusion/ej2-react-charts';
import { stripData } from 'datasource.ts';
function App() {
    const chartData: any[] = [{x: 1, y: 5},{x: 2, y: 39},{x: 3, y: 21},{x: 4,
y: 51},{x: 5, y: 30},
                                {x: 6, y: 25},{x: 7, y: 10},{x: 8, y: 40},{x:
9, y: 50},{x: 10, y: 20}];
    const primaryyAxis: AxisModel = { stripLines:[
        {start: 20, end: 40, isSegmented: true, segmentStart: 2,
segmentEnd: 4,
        color: 'rgba(167,169,171, 0.3)'}
    ]
    };
    const marker = { visible: true };
    return <ChartComponent id='charts'
        primaryYAxis={primaryyAxis}>
        <Inject services={[ColumnSeries,LineSeries, Legend, Tooltip,
DataLabel, Category, StripLine]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} xName='x' yName='y'
type='Line' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
    };
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));
}

```

See Also

- [Mark the threshold in chart](#)

Multiple panes in React Chart component

Chart area can be divided into multiple panes using [rows](#) and [columns](#).

Rows

To split the chart area vertically into number of rows, use [rows](#) property of the chart.

*1. You can allocate space for each row by using the [height](#) property.

The value can be either in percentage or in pixel.*

2. To associate a vertical axis to a particular row, specify its index to [rowIndex](#) property of the axis.

3. To customize each row's bottom line, use [border](#) property.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, RowsDirective,
RowDirective, AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, LineSeries } from
"@syncfusion/ej2-react-charts";
function App() {
    const data = [
        { x: 'Jan', y: 15, y1: 33 }, { x: 'Feb', y: 20, y1: 31 }, { x:
'Mar', y: 35, y1: 30 },
        { x: 'Apr', y: 40, y1: 28 }, { x: 'May', y: 80, y1: 29 }, { x:
'Jun', y: 70, y1: 30 },
        { x: 'Jul', y: 65, y1: 33 }, { x: 'Aug', y: 55, y1: 32 }, { x:
'Sep', y: 50, y1: 34 },
        { x: 'Oct', y: 30, y1: 32 }, { x: 'Nov', y: 35, y1: 32 }, { x:
'Dec', y: 35, y1: 31 }
    ];
    const primaryxAxis = { valueType: 'Category', title: 'Months', interval:
1 };
    const primaryyAxis = {
        title: 'Temperature (Fahrenheit)', minimum: 0, maximum: 90,
interval: 20,
        lineStyle: { width: 0 }, labelFormat: '{value}°F'
    };
    const lines = { width: 0 };
    const marker = { visible: true, width: 10, height: 10, border: { width:
2, color: '#F8AB1D' } };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Weather Condition'>
        <Inject services={[ColumnSeries, LineSeries, Legend, Tooltip,
DataLabel, Category]}/>
        <AxesDirective>
            <AxisDirective rowIndex={1} name='yAxis1' opposedPosition={true}
title='Temperature (Celsius)' labelFormat='{value}°C' minimum={24}
maximum={36} interval={2} majorGridLines={lines} lineStyle={lines}>
            </AxisDirective>
        </AxesDirective>
        <RowsDirective>
            <RowDirective height='50%'></RowDirective>
            <RowDirective height='50%'></RowDirective>
        </RowsDirective>
        <SeriesCollectionDirective>
```

```

        <SeriesDirective dataSource={data} xName='x' yName='y'
name='Germany' type='Column'>
        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y1' name='Japan'
type='Line' marker={marker} yAxisName='yAxis1'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
RowsDirective, RowDirective, AxesDirective, AxisDirective, SeriesDirective,
Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { x: 'Jan', y: 15, y1: 33 }, { x: 'Feb', y: 20, y1: 31 }, { x: 'Mar', y:
35, y1: 30 },
        { x: 'Apr', y: 40, y1: 28 }, { x: 'May', y: 80, y1: 29 }, { x: 'Jun', y:
70, y1: 30 },
        { x: 'Jul', y: 65, y1: 33 }, { x: 'Aug', y: 55, y1: 32 }, { x: 'Sep', y:
50, y1: 34 },
        { x: 'Oct', y: 30, y1: 32 }, { x: 'Nov', y: 35, y1: 32 }, { x: 'Dec', y:
35, y1: 31 }
    ];
    const primaryxAxis: AxisModel = { valueType: 'Category', title: 'Months',
interval: 1 };
    const primaryyAxis: AxisModel = {
        title: 'Temperature (Fahrenheit)', minimum: 0, maximum: 90, interval:
20,
        lineStyle: { width: 0 }, labelFormat: '{value}°F'
    };
    const lines = { width: 0 };
    const marker = { visible: true, width: 10, height: 10, border: { width: 2,
color: '#F8AB1D' } };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Weather Condition'>
        <Inject services={[ColumnSeries, LineSeries, Legend, Tooltip,
DataLabel, Category]} />
        <AxesDirective>
            <AxisDirective rowIndex={1} name='yAxis1' opposedPosition={true}
title='Temperature (Celsius)'
                labelFormat='{value}°C' minimum={24} maximum={36} interval={2}
                majorGridLines={lines} lineStyle={lines} >
            </AxisDirective>

```

```

    </AxesDirective>
    <RowsDirective>
      <RowDirective height='50%' ></RowDirective>
      <RowDirective height='50%' ></RowDirective>
    </RowsDirective>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y'
name='Germany' type='Column'>
    </SeriesDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y1' name='Japan'
type='Line'
      marker={marker} yAxisName='yAxis1' >
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

For spanning the vertical axis along multiple row, you can use [span](#) property of an axis.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, ColumnSeries, Legend, Category,
Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-react-charts';
function App() {
  const data = [
    { x: 'Jan', y: 15, y1: 33 }, { x: 'Feb', y: 20, y1: 31 }, { x:
'Mar', y: 35, y1: 30 },
    { x: 'Apr', y: 40, y1: 28 }, { x: 'May', y: 80, y1: 29 }, { x:
'Jun', y: 70, y1: 30 },
    { x: 'Jul', y: 65, y1: 33 }, { x: 'Aug', y: 55, y1: 32 }, { x:
'Sep', y: 50, y1: 34 },
    { x: 'Oct', y: 30, y1: 32 }, { x: 'Nov', y: 35, y1: 32 }, { x:
'Dec', y: 35, y1: 31 }
  ];
  const primaryxAxis = { valueType: 'Category', title: 'Months', interval:
1 };
  const primaryyAxis = {
    title: 'Temperature (Fahrenheit)', minimum: 0, maximum: 90,
interval: 20,
    lineStyle: { width: 0 }, labelFormat: '{value}°F'
  };
  const lines = { width: 0 };
  const marker = { visible: true, width: 10, height: 10, border: { width:
2, color: '#F8AB1D' } };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Weather Condition'>
    <Inject services={[ColumnSeries, LineSeries, Legend, Tooltip,
DataLabel, Category]} />
    <AxesDirective>

```



```

    <AxisDirective rowIndex={1} name='yAxis1' opposedPosition={true}
    title='Temperature (Celsius)' labelFormat='{value}°C' minimum={24}
    maximum={36} interval={2} majorGridLines={lines} lineStyle={lines}>
    </AxisDirective>
  </AxesDirective>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y'
    name='Germany' type='Column'>
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y1' name='Japan'
    type='Line' marker={marker} yAxisName='yAxis1'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { x: 'Jan', y: 15, y1: 33 }, { x: 'Feb', y: 20, y1: 31 }, { x: 'Mar', y:
35, y1: 30 },
    { x: 'Apr', y: 40, y1: 28 }, { x: 'May', y: 80, y1: 29 }, { x: 'Jun', y:
70, y1: 30 },
    { x: 'Jul', y: 65, y1: 33 }, { x: 'Aug', y: 55, y1: 32 }, { x: 'Sep', y:
50, y1: 34 },
    { x: 'Oct', y: 30, y1: 32 }, { x: 'Nov', y: 35, y1: 32 }, { x: 'Dec', y:
35, y1: 31 }
  ];
  const primaryXAxis: AxisModel = { valueType: 'Category', title: 'Months',
interval: 1 };
  const primaryYAxis: AxisModel = {
    title: 'Temperature (Fahrenheit)', minimum: 0, maximum: 90, interval:
20,
    lineStyle: { width: 0 }, labelFormat: '{value}°F'
  };
  const lines = { width: 0 };
  const marker = { visible: true, width: 10, height: 10, border: { width: 2,
color: '#F8AB1D' } };
  return <ChartComponent id='charts'
    primaryXAxis={primaryXAxis}
    primaryYAxis={primaryYAxis}
    title='Weather Condition'>
    <Inject services={[ColumnSeries, LineSeries, Legend, Tooltip,
DataLabel, Category]} />
    <AxesDirective>

```

```

    <AxisDirective rowIndex={1} name='yAxis1' opposedPosition={true}
    title='Temperature (Celsius)'
      labelFormat='{value}°C' minimum={24} maximum={36} interval={2}
      majorGridLines={lines} lineStyle={lines} >
    </AxisDirective>
  </AxesDirective>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y'
    name='Germany' type='Column'>
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y1' name='Japan'
    type='Line'
      marker={marker} yAxisName='yAxis1' >
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Columns

To split the chart area horizontally into number of columns, use [columns](#) property of the chart.

*1. You can allocate space for each column by using the [width](#)

property. The given width can be either in percentage or in pixel.*

*2. To associate a horizontal axis to a particular column, specify its index to

[columnIndex](#) property of the axis.*

*3. To customize each column's bottom line, use [border](#)

property.*

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, ColumnsDirective,
ColumnDirective, AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, LineSeries } from
"@syncfusion/ej2-react-charts";
function App() {
  const data = [
    { x: 'Jan', y: 15, y1: 33 }, { x: 'Feb', y: 20, y1: 31 }, { x:
'Mar', y: 35, y1: 30 },
    { x: 'Apr', y: 40, y1: 28 }, { x: 'May', y: 80, y1: 29 }, { x:
'Jun', y: 70, y1: 30 },
    { x: 'Jul', y: 65, y1: 33 }, { x: 'Aug', y: 55, y1: 32 }, { x:
'Sep', y: 50, y1: 34 },
    { x: 'Oct', y: 30, y1: 32 }, { x: 'Nov', y: 35, y1: 32 }, { x:
'Dec', y: 35, y1: 31 }
  ];
  const primaryxAxis = { valueType: 'Category', title: 'Months', interval:
1 };
  const primaryyAxis = {

```

```

        title: 'Temperature (Fahrenheit)', minimum: 0, maximum: 90,
        interval: 20,
        lineStyle: { width: 0 }, labelFormat: '{value}°F'
    };
    const lines = { width: 0 };
    const marker = { visible: true, width: 10, height: 10, border: { width:
2, color: '#F8AB1D' } };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Weather Condition'>
        <Inject services={[ColumnSeries, LineSeries, Legend, Tooltip,
DataLabel, Category]}/>
        <AxesDirective>
            <AxisDirective columnIndex={1} name='xAxis1' opposedPosition={true}
valueType='Category' majorGridLines={lines} lineStyle={lines}>
                </AxisDirective>
            </AxesDirective>
            <ColumnsDirective>
                <ColumnDirective width='50%'></ColumnDirective>
                <ColumnDirective width='50%'></ColumnDirective>
            </ColumnsDirective>
            <SeriesCollectionDirective>
                <SeriesDirective dataSource={data} xName='x' yName='y'
name='Germany' type='Column'>
                    </SeriesDirective>
                <SeriesDirective dataSource={data} xName='x' yName='y1' name='Japan'
type='Line' marker={marker} xAxisName='xAxis1'>
                    </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
    }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
ColumnsDirective, ColumnDirective, AxesDirective, AxisDirective,
SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { x: 'Jan', y: 15, y1: 33 }, { x: 'Feb', y: 20, y1: 31 }, { x: 'Mar', y:
35, y1: 30 },
        { x: 'Apr', y: 40, y1: 28 }, { x: 'May', y: 80, y1: 29 }, { x: 'Jun', y:
70, y1: 30 },
        { x: 'Jul', y: 65, y1: 33 }, { x: 'Aug', y: 55, y1: 32 }, { x: 'Sep', y:
50, y1: 34 },
        { x: 'Oct', y: 30, y1: 32 }, { x: 'Nov', y: 35, y1: 32 }, { x: 'Dec', y:
35, y1: 31 }
    ];

```

```

const primaryxAxis: AxisModel = { valueType: 'Category', title: 'Months',
interval: 1 };
const primaryyAxis: AxisModel = {
  title: 'Temperature (Fahrenheit)', minimum: 0, maximum: 90, interval:
20,
  lineStyle: { width: 0 }, labelFormat: '{value}°F'
};
const lines = { width: 0 };
const marker = { visible: true, width: 10, height: 10, border: { width: 2,
color: '#F8AB1D' } };
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  title='Weather Condition'>
  <Inject services={[ColumnSeries, LineSeries, Legend, Tooltip,
DataLabel, Category]} />
  <AxesDirective>
    <AxisDirective columnIndex={1} name='xAxis1' opposedPosition={true}
valueType='Category'
      majorGridLines={lines} lineStyle={lines} >
    </AxisDirective>
  </AxesDirective>
  <ColumnsDirective>
    <ColumnDirective width='50%' ></ColumnDirective>
    <ColumnDirective width='50%' ></ColumnDirective>
  </ColumnsDirective>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y'
name='Germany' type='Column'>
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y1' name='Japan'
type='Line'
      marker={marker} xAxisName='xAxis1' >
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

For spanning the horizontal axis along multiple column, you can use [span](#) property of an axis.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, ColumnsDirective,
ColumnDirective, AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, LineSeries } from
'@syncfusion/ej2-react-charts';
function App() {
  const data = [
    { x: 'Jan', y: 15, y1: 33 }, { x: 'Feb', y: 20, y1: 31 }, { x:
'Mar', y: 35, y1: 30 },
    { x: 'Apr', y: 40, y1: 28 }, { x: 'May', y: 80, y1: 29 }, { x:
'Jun', y: 70, y1: 30 },

```

```

    { x: 'Jul', y: 65, y1: 33 }, { x: 'Aug', y: 55, y1: 32 }, { x:
'Sep', y: 50, y1: 34 },
    { x: 'Oct', y: 30, y1: 32 }, { x: 'Nov', y: 35, y1: 32 }, { x:
'Dec', y: 35, y1: 31 }
  ];
  const primaryxAxis = { valueType: 'Category', span: 2, title: 'Months',
interval: 1 };
  const primaryyAxis = {
    title: 'Temperature (Fahrenheit)', minimum: 0, maximum: 90,
interval: 20,
    lineStyle: { width: 0 }, labelFormat: '{value}°F'
  };
  const lines = { width: 0 };
  const marker = { visible: true, width: 10, height: 10, border: { width:
2, color: '#F8AB1D' } };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Weather Condition'>
    <Inject services={[ColumnSeries, LineSeries, Legend, Tooltip,
DataLabel, Category]}/>
    <AxesDirective>
      <AxisDirective columnIndex={1} name='xAxis1' opposedPosition={true}
valueType='Category' majorGridLines={lines} lineStyle={lines}>
        </AxisDirective>
      </AxesDirective>
      <ColumnsDirective>
        <ColumnDirective width='50%'></ColumnDirective>
        <ColumnDirective width='50%'></ColumnDirective>
      </ColumnsDirective>
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y'
name='Germany' type='Column'>
          </SeriesDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y1' name='Japan'
type='Line' marker={marker} xAxisName='xAxis1'>
          </SeriesDirective>
        </SeriesCollectionDirective>
      </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
ColumnsDirective, ColumnDirective, AxesDirective, AxisDirective,
SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { x: 'Jan', y: 15, y1: 33 }, { x: 'Feb', y: 20, y1: 31 }, { x: 'Mar', y:
35, y1: 30 },

```

```

    { x: 'Apr', y: 40, y1: 28 }, { x: 'May', y: 80, y1: 29 }, { x: 'Jun', y:
70, y1: 30 },
    { x: 'Jul', y: 65, y1: 33 }, { x: 'Aug', y: 55, y1: 32 }, { x: 'Sep', y:
50, y1: 34 },
    { x: 'Oct', y: 30, y1: 32 }, { x: 'Nov', y: 35, y1: 32 }, { x: 'Dec', y:
35, y1: 31 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category', span: 2, title:
'Months', interval: 1 };
  const primaryyAxis: AxisModel = {
    title: 'Temperature (Fahrenheit)', minimum: 0, maximum: 90, interval:
20,
    lineStyle: { width: 0 }, labelFormat: '{value}°F'
  };
  const lines = { width: 0 };
  const marker = { visible: true, width: 10, height: 10, border: { width: 2,
color: '#F8AB1D' } };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Weather Condition'>
    <Inject services=[ColumnSeries, LineSeries, Legend, Tooltip,
DataLabel, Category] />
    <AxesDirective>
      <AxisDirective columnIndex={1} name='xAxis1' opposedPosition={true}
valueType='Category'
        majorGridLines={lines} lineStyle={lines} >
      </AxisDirective>
    </AxesDirective>
    <ColumnsDirective>
      <ColumnDirective width='50%' ></ColumnDirective>
      <ColumnDirective width='50%' ></ColumnDirective>
    </ColumnsDirective>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y'
name='Germany' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y1' name='Japan'
type='Line'
        marker={marker} xAxisName='xAxis1' >
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));

```

Chart Types

Line Chart in React Chart component

<!-- markdownlint-disable MD036 -->

Line

To render a line series, use series [type](#) as `Line` and inject `LineSeries` module into the `services`.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
import { data } from 'datasource.ts';
function App() {
    return <ChartComponent id='charts'>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India' type='Line'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries,
Selection}
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
    return <ChartComponent id='charts'>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category]}
/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India' type='Line'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Multicolored Line

To render a multicolored line series, use series [type](#) as `Line` and inject `LineSeries` module into the services.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
```

```
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, MultiColoredLineSeries, LineSeries }
from '@syncfusion/ej2-react-charts';
function App() {
    const data = [{ x: 2005, y: 28, color: 'red' }, { x: 2006, y: 25, color:
'green' },
        { x: 2007, y: 26, color: '#ff0097' }, { x: 2008, y: 27, color:
'crimson' },
        { x: 2009, y: 32, color: 'blue' }, { x: 2010, y: 35, color:
'darkorange' }];
    return <ChartComponent id='charts'>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category,
MultiColoredLineSeries]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India' type='MultiColoredLine' pointColorMapping='color'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, MultiColoredLineSeries,
LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [{ x: 2005, y: 28, color: 'red' }, { x: 2006, y: 25,
color: 'green' },
        { x: 2007, y: 26, color: '#ff0097' }, { x: 2008, y:
27, color: 'crimson' },
        { x: 2009, y: 32, color: 'blue' }, { x: 2010, y: 35,
color: 'darkorange' }];
    return <ChartComponent id='charts'>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category,
MultiColoredLineSeries]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India' type='MultiColoredLine'
            pointColorMapping='color'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Series customization

The following properties can be used to customize the **line** series.

- [fill](#) – Specifies the color of the series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes for series.
- [width](#) – Specifies the width for series.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
import { data } from 'datasource.ts';
function App() {
    const marker = { visible: true, width: 10, height: 10, border: { width:
2, color: '#F8AB1D' } };
    return <ChartComponent id='charts'>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' fill='green'
width={3} dashArray='5,5' name='India' type='Line' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries,
Selection}
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
    const marker = { visible: true, width: 10, height: 10, border: { width: 2,
color: '#F8AB1D' } };
    return <ChartComponent id='charts'>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category]}
/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' fill='green'
width={3} dashArray='5,5'
            name='India' type='Line'
            marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
```

```
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

See Also

- [Data label](#)
- [Tooltip](#)

Step line in React Chart component

Step line

To render a step line series, use series [type](#) as `StepLine` and inject `StepLineSeries` module into the services.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StepLineSeries } from
"@syncfusion/ej2-react-charts";
import { steplineData } from "datasource.ts";
function App() {
    const marker = { visible: true, width: 10, height: 10, border: { width:
2, color: '#F8AB1D' } };
    return <ChartComponent id='charts'>
        <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={steplineData} xName='x' yName='y'
width={2} name='India' type='StepLine' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
StepLineSeries, Selection}
from "@syncfusion/ej2-react-charts";
import { steplineData } from "datasource.ts";
function App() {
    const marker = { visible: true, width: 10, height: 10, border: { width: 2,
color: '#F8AB1D' } };
    return <ChartComponent id='charts'>
        <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
Category]} />
        <SeriesCollectionDirective>
```

```

        <SeriesDirective dataSource={steplineData} xName='x' yName='y'
width={2} name='India' type='StepLine'
        marker={marker}>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Series customization

The following properties can be used to customize the **step line** series.

- [fill](#) – Specifies the color of the series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes for series.
- [width](#) – Specifies the width for series.
- [step](#) – Specifies the position of the step for the series.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StepLineSeries } from
'@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
    const marker = { visible: true, width: 10, height: 10, border: { width:
2, color: '#F8AB1D' } };
    return <ChartComponent id='charts'>
        <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' fill='green'
width={3} dashArray='5,5' type='StepLine' marker={marker} opacity={0.5}
step='Left'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
StepLineSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';

```

```
function App() {
  const marker = { visible: true, width: 10, height: 10, border: { width: 2,
color: '#F8AB1D' } };
  return <ChartComponent id='charts'>
    <Inject services=[StepLineSeries, Legend, Tooltip, DataLabel,
Category] />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' fill='green'
width={3} dashArray='5,5'
      type='StepLine' opacity={0.5} step='Left' marker={marker}>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

See also

- [Data label](#)
- [Tooltip](#)

Stacked Line in React Chart component

Stacked Line

To render a stacked line series, use series [type](#) as `StackingLine` and inject `StackingLineSeries` module into the `services`.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StackingLineSeries } from
'@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const marker = { visible: true, width: 10, height: 10, border: { width:
2, color: '#F8AB1D' } };
  return <ChartComponent id='charts' primaryXAxis={{ interval: 1,
valueType: 'Category' }} primaryYAxis={{ title: 'Expense', minimum: 0,
maximum: 400, interval: 100, labelFormat: '$ {value}' }}>
    <Inject services=[StackingLineSeries, Legend, Tooltip, DataLabel,
Category] />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='x'
yName='y' name='John' width='2' type='StackingLine' marker={{ visible: true
}} dashArray='5,1'>
    </SeriesDirective>
      <SeriesDirective dataSource={chartData} xName='x'
yName='y1' name='Peter' width='2' type='StackingLine' marker={{ visible:
true }} dashArray='5,1'>
    </SeriesDirective>
  </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

```

        <SeriesDirective dataSource={chartData} xName='x'
yName='y2' name='Steve' width='2' type='StackingLine' marker={{ visible:
true }} dashArray='5,1'>
        </SeriesDirective>
        <SeriesDirective dataSource={chartData} xName='x'
yName='y3' name='Charle' width='2' type='StackingLine' marker={{ visible:
true }} dashArray='5,1'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
    Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
    StackingLineSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const marker = { visible: true, width: 10, height: 10, border: { width: 2,
color: '#F8AB1D' } };
    return <ChartComponent id='charts'
        primaryXAxis={{ interval: 1, valueType: 'Category' }}
        primaryYAxis={{ title: 'Expense', minimum: 0, maximum: 400,
interval: 100, labelFormat: '${value}' }}>
        <Inject services={[StackingLineSeries, Legend, Tooltip, DataLabel,
Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} xName='x'
yName='y' name='John' width='2' type='StackingLine' marker= {{visible:
true}} dashArray='5,1'>
            </SeriesDirective>
            <SeriesDirective dataSource={chartData} xName='x'
yName='y1' name='Peter' width='2' type='StackingLine' marker= {{visible:
true}} dashArray='5,1'>
            </SeriesDirective>
            <SeriesDirective dataSource={chartData} xName='x'
yName='y2' name='Steve' width='2' type='StackingLine' marker= {{visible:
true}} dashArray='5,1'>
            </SeriesDirective>
            <SeriesDirective dataSource={chartData} xName='x'
yName='y3' name='Charle' width='2' type='StackingLine' marker= {{visible:
true}} dashArray='5,1'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;

```

```
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

Series customization

The following properties can be used to customize the **stacked line** series.

- [fill](#) – Specifies the color of the series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes for series.
- [width](#) – Specifies the width for series.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StackingLineSeries } from
"@syncfusion/ej2-react-charts";
import { chartData } from "datasource.ts";
function App() {
    const marker = { visible: true, width: 10, height: 10, border: { width:
2, color: '#F8AB1D' } };
    return <ChartComponent id='charts' primaryXAxis={{ interval: 1,
valueType: 'Category' }} primaryYAxis={{ title: 'Expense', minimum: 0,
maximum: 400, interval: 100, labelFormat: '${value}' }}>
        <Inject services={[StackingLineSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} xName='x'
yName='y' name='John' width='2' type='StackingLine' fill='blue' marker={{
visible: true }} dashArray='5,1'>
            </SeriesDirective>
            <SeriesDirective dataSource={chartData} xName='x'
yName='y1' name='Peter' width='2' type='StackingLine' fill='brown' marker={{
visible: true }} dashArray='5,1'>
            </SeriesDirective>
            <SeriesDirective dataSource={chartData} xName='x'
yName='y2' name='Steve' width='2' type='StackingLine' fill='yellow'
marker={{ visible: true }} dashArray='5,1'>
            </SeriesDirective>
            <SeriesDirective dataSource={chartData} xName='x'
yName='y3' name='Charle' width='2' type='StackingLine' fill='grey' marker={{
visible: true }} dashArray='5,1'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
      Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
      StackingLineSeries, Selection }
from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const marker = { visible: true, width: 10, height: 10, border: { width: 2,
    color: '#F8AB1D' } };
  return <ChartComponent id='charts'
    primaryXAxis={{ interval: 1, valueType: 'Category' }}
    primaryYAxis={{ title: 'Expense', minimum: 0, maximum: 400,
    interval: 100, labelFormat: '${value}' }}>
    <Inject services={[StackingLineSeries, Legend, Tooltip, DataLabel,
    Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='x'
      yName='y' name='John' width='2' type='StackingLine' marker= {{visible:
      true}} fill='blue' dashArray='5,1'>
      </SeriesDirective>
      <SeriesDirective dataSource={chartData} xName='x'
      yName='y1' name='Peter' width='2' type='StackingLine' marker= {{visible:
      true}} fill='brown' dashArray='5,1'>
      </SeriesDirective>
      <SeriesDirective dataSource={chartData} xName='x'
      yName='y2' name='Steve' width='2' type='StackingLine' marker= {{visible:
      true}} fill='yellow' dashArray='5,1'>
      </SeriesDirective>
      <SeriesDirective dataSource={chartData} xName='x'
      yName='y3' name='Charle' width='2' type='StackingLine' marker= {{visible:
      true}} fill='grey' dashArray='5,1'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

See Also

- [Data label](#)
- [Tooltip](#)

100% Stacked Line in React Chart component

100% Stacked Line

To render a 100% stacked line series, use series [type](#) as `StackingLine100` and inject `StackingLineSeries` module into the `services`.

INDEX.JSX

```
{% raw %}
```

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StackingLineSeries } from
'@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const marker = { visible: true, width: 10, height: 10, border: { width:
2, color: '#F8AB1D' } };
    return <ChartComponent id='charts' primaryXAxis={{ interval: 1,
valueType: 'Category' }} primaryYAxis={{ title: 'Expense', interval: 20 }}
title='Family Expense for Month'>
        <Inject services={[StackingLineSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} xName='x' yName='y'
name='John' width='2' type='StackingLine100' marker={{ visible: true }}
dashArray='5,1'>
            </SeriesDirective>
            <SeriesDirective dataSource={chartData} xName='x' yName='y1'
name='Peter' width='2' type='StackingLine100' marker={{ visible: true }}
dashArray='5,1'>
            </SeriesDirective>
            <SeriesDirective dataSource={chartData} xName='x' yName='y2'
name='Steve' width='2' type='StackingLine100' marker={{ visible: true }}
dashArray='5,1'>
            </SeriesDirective>
            <SeriesDirective dataSource={chartData} xName='x' yName='y3'
name='Charle' width='2' type='StackingLine100' marker={{ visible: true }}
dashArray='5,1'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
StackingLineSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const marker = { visible: true, width: 10, height: 10, border: { width: 2,
color: '#F8AB1D' } };
    return <ChartComponent id='charts'
        primaryXAxis={{ interval: 1, valueType: 'Category' }}
        primaryYAxis={{ title: 'Expense', interval: 20 }}
        title='Family Expense for Month'>

```



```

    <Inject services={[StackingLineSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='x' yName='y'
name='John' width='2' type='StackingLine100' marker= {{visible: true}}
dashArray='5,1'>
      </SeriesDirective>
      <SeriesDirective dataSource={chartData} xName='x' yName='y1'
name='Peter' width='2' type='StackingLine100' marker= {{visible: true}}
dashArray='5,1'>
      </SeriesDirective>
      <SeriesDirective dataSource={chartData} xName='x' yName='y2'
name='Steve' width='2' type='StackingLine100' marker= {{visible: true}}
dashArray='5,1'>
      </SeriesDirective>
      <SeriesDirective dataSource={chartData} xName='x' yName='y3'
name='Charle' width='2' type='StackingLine100' marker= {{visible: true}}
dashArray='5,1'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Series customization

The following properties can be used to customize the **100% stacked line** series.

- [fill](#) – Specifies the color of the series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes for series.
- [width](#) – Specifies the width for series.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StackingLineSeries } from
'@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const marker = { visible: true, width: 10, height: 10, border: { width:
2, color: '#F8AB1D' } };
  return <ChartComponent id='charts' primaryXAxis={{ interval: 1,
valueType: 'Category' }} primaryYAxis={{ title: 'Expense', interval: 20 }}
title='Family Expense for Month'>
    <Inject services={[StackingLineSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>

```

```

        <SeriesDirective dataSource={chartData} xName='x' yName='y'
name='John' width='2' type='StackingLine100' marker={{ visible: true }}
dashArray='5,1' fill='blue'>
        </SeriesDirective>
        <SeriesDirective dataSource={chartData} xName='x' yName='y1'
name='Peter' width='2' type='StackingLine100' marker={{ visible: true }}
dashArray='5,1' fill='yellow'>
        </SeriesDirective>
        <SeriesDirective dataSource={chartData} xName='x' yName='y2'
name='Steve' width='2' type='StackingLine100' marker={{ visible: true }}
dashArray='5,1' fill='red'>
        </SeriesDirective>
        <SeriesDirective dataSource={chartData} xName='x' yName='y3'
name='Charle' width='2' type='StackingLine100' marker={{ visible: true }}
dashArray='5,1' fill='grey'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
    Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
    StackingLineSeries, Selection }
from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const marker = { visible: true, width: 10, height: 10, border: { width: 2,
    color: '#F8AB1D' } };
    return <ChartComponent id='charts'
        primaryXAxis={{ interval: 1, valueType: 'Category' }}
        primaryYAxis={{ title: 'Expense', interval: 20 }}
        title='Family Expense for Month'>
        <Inject services={[StackingLineSeries, Legend, Tooltip, DataLabel,
    Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} xName='x' yName='y'
name='John' width='2' type='StackingLine100' marker= {{visible: true}}
dashArray='5,1' fill='blue'>
            </SeriesDirective>
            <SeriesDirective dataSource={chartData} xName='x' yName='y1'
name='Peter' width='2' type='StackingLine100' marker= {{visible: true}}
dashArray='5,1' fill='yellow'>
            </SeriesDirective>
            <SeriesDirective dataSource={chartData} xName='x' yName='y2'
name='Steve' width='2' type='StackingLine100' marker= {{visible: true}}
dashArray='5,1' fill='red'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>

```

```

        <SeriesDirective dataSource={chartData} xName='x' yName='y3'
name='Charle' width='2' type='StackingLine100' marker= {{visible: true}}
dashArray='5,1' fill='grey'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

See Also

- [Data label](#)
- [Tooltip](#)

Spline in React Chart component

Spline

To render a spline series, use series [type](#) as **Spline** and inject **SplineSeries** module into the **services**.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, SplineSeries } from '@syncfusion/ej2-
react-charts';
import { splineData } from 'datasource.ts';
function App() {
    const primaryxAxis = { title: 'Month', valueType: 'Category' };
    const primaryyAxis = {
        minimum: -5, maximum: 35, interval: 5,
        title: 'Temperature in Celsius', labelFormat: '{value}C'
    };
    const marker = { visible: true, width: 10, height: 10 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='CO2 - Intensity Analysis'>
        <Inject services={[SplineSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={splineData} xName='x' yName='y'
width={2} name='London' type='Spline' marker={marker}>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
    }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
        Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
SplineSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { splineData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { title: 'Month', valueType: 'Category' };
    const primaryyAxis: AxisModel = {
        minimum: -5, maximum: 35, interval: 5,
        title: 'Temperature in Celsius', labelFormat: '{value}C'
    };
    const marker = { visible: true, width: 10, height: 10 };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='CO2 - Intensity Analysis'>
        <Inject services={[SplineSeries, Legend, Tooltip, DataLabel,
Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={splineData} xName='x' yName='y'
width={2} name='London' type='Spline'
                marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
    </>
    export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

Series customization

The following properties can be used to customize the `spline` series.

- `fill` – Specifies the color of the series.
- `opacity` – Specifies the opacity of `fill`.
- `dashArray` – Specifies the dashes for series.
- `width` – Specifies the width for series.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, SplineSeries } from '@syncfusion/ej2-
react-charts';
import { data } from 'datasource.ts';
function App() {
    const marker = { visible: true, width: 10, height: 10, border: { width:
2, color: '#F8AB1D' } };
    return <ChartComponent id='charts'>
```

```

    <Inject services={[SplineSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' fill='green'
width={3} dashArray='5,5' name='India' type='Spline' marker={marker}>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
SplineSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
  const marker = { visible: true, width: 10, height: 10, border: { width: 2,
color: '#F8AB1D' } };
  return <ChartComponent id='charts'>
    <Inject services={[SplineSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' fill='green'
width={3} dashArray='5,5'
        name='India' type='Spline'
        marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

See Also

- [Data label](#)
- [Tooltip](#)

Area series in React Chart component

Area

To render a area series, use series [type](#) as `Area` and inject `AreaSeries` module into the `services`.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";

```

```

import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, AreaSeries } from '@syncfusion/ej2-
react-charts';
function App() {
  const data = [
    { x: 1900, y: 4 }, { x: 1920, y: 3.0 }, { x: 1940, y: 3.8 },
    { x: 1960, y: 3.4 }, { x: 1980, y: 3.2 }, { x: 2000, y: 3.9 }
  ];
  const primaryxAxis = {
    title: 'Year', minimum: 1900, maximum: 2000, interval: 10,
    edgeLabelPlacement: 'Shift'
  };
  const primaryyAxis = { minimum: 2, maximum: 5, interval: 0.5, title:
'Sales Amount in Millions' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Average Sales Comparison'>
    <Inject services={[AreaSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' name='Product
A' fill='#69D2E7' opacity={0.6} type='Area'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, AreaSeries,
Selection}
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { x: 1900, y: 4 }, { x: 1920, y: 3.0 }, { x: 1940, y: 3.8 },
    { x: 1960, y: 3.4 }, { x: 1980, y: 3.2 }, { x: 2000, y: 3.9 }
  ];
  const primaryxAxis: AxisModel = {
    title: 'Year', minimum: 1900, maximum: 2000, interval: 10,
    edgeLabelPlacement: 'Shift';
  }
  const primaryyAxis: AxisModel = { minimum: 2, maximum: 5, interval: 0.5,
title: 'Sales Amount in Millions' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Average Sales Comparison'>
    <Inject services={[AreaSeries, Legend, Tooltip, DataLabel, Category]}
/>
    <SeriesCollectionDirective>

```

```

    <SeriesDirective dataSource={data} xName='x' yName='y' name='Product
A' fill='#69D2E7'
      opacity={0.6} type='Area'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Multicolored area

To render a multicolored area series, use the series type as `MultiColoredArea`, and inject the `MultiColoredAreaSeries` module into the services. The required segments of the series can be customized using the `value`, `color`, and `dashArray`.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
MultiColoredAreaSeries, SegmentsDirective, Legend, StepAreaSeries,
SegmentDirective } from '@syncfusion/ej2-react-charts';
function App() {
  const data = [{ x: 2005, y: 28 }, { x: 2006, y: 25 }, { x: 2007, y: 26
}, { x: 2008, y: 27 },
    { x: 2009, y: 32 }, { x: 2010, y: 35 }, { x: 2011, y: 25 }];
  return <ChartComponent id='charts'>
    <Inject services={[StepAreaSeries, Legend, MultiColoredAreaSeries]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y'
name='England' type='MultiColoredArea' segmentAxis='X'>
        <SegmentsDirective>
          <SegmentDirective value={2007} color='blue'></SegmentDirective>
          <SegmentDirective value={2009} color='green'></SegmentDirective>
          <SegmentDirective color='orange'></SegmentDirective>
        </SegmentsDirective>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
MultiColoredAreaSeries, SegmentsDirective,
  Legend, StepAreaSeries, SegmentDirective }
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [{ x: 2005, y: 28 }, { x: 2006, y: 25 }, { x: 2007, y:
26 }, { x: 2008, y: 27 },

```

```

        { x: 2009, y: 32 }, { x: 2010, y: 35 }, { x: 2011,
y: 25 }]]
    return <ChartComponent id='charts'>
        <Inject services={[StepAreaSeries, Legend, MultiColoredAreaSeries]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y'
name='England' type='MultiColoredArea' segmentAxis='X' >
                <SegmentsDirective>
                    <SegmentDirective value={2007} color='blue'></SegmentDirective>
                    <SegmentDirective value={2009} color='green'></SegmentDirective>
                    <SegmentDirective color='orange'></SegmentDirective>
                </SegmentsDirective>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Series customization

The following properties can be used to customize the **area** series.

- [fill](#) – Specifies the color of the area series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes of series.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, AreaSeries } from '@syncfusion/ej2-
react-charts';
import { areaData } from 'datasource.ts';
function App() {
    const primaryxAxis = {
        title: 'Year', minimum: 1900, maximum: 2000, interval: 10,
        edgeLabelPlacement: 'Shift'
    };
    const primaryyAxis = { minimum: 2, maximum: 5, interval: 0.5, title:
'Sales Amount in Millions' };
    const border = { color: 'red', width: 2 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Average Sales Comparison'>
        <Inject services={[AreaSeries, Legend, Tooltip, DataLabel,
Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={areaData} xName='x' yName='y'
name='Product A' fill='green' width={2} dashArray='5,5' border={border}
opacity={0.6} type='Area'>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
}
;

```



```
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, AreaSeries,
Selection}
from '@syncfusion/ej2-react-charts';
import { areaData } from 'datasource.ts';
function App() {
  const primaryXAxis: AxisModel = {
    title: 'Year', minimum: 1900, maximum: 2000, interval: 10,
    edgeLabelPlacement: 'Shift'
  };
  const primaryYAxis: AxisModel = { minimum: 2, maximum: 5, interval: 0.5,
title: 'Sales Amount in Millions' };
  const border = { color: 'red', width: 2 };
  return <ChartComponent id='charts'
    primaryXAxis={primaryXAxis}
    primaryYAxis={primaryYAxis}
    title='Average Sales Comparison'>
    <Inject services={[AreaSeries, Legend, Tooltip, DataLabel, Category]}
  />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={areaData} xName='x' yName='y'
name='Product A'
        fill='green' width={2} dashArray='5,5' border={border}
        opacity={0.6} type='Area'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Area border

The following properties in the [bordermodel](#) can be used to customize the border of the Area Chart.

- [width](#) - Specifies the width for the border of the Area Chart.
- [color](#) - Specifies the Color for the border of the Area Chart.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, Legend, Category, Tooltip, DataLabel, AreaSeries }
from '@syncfusion/ej2-react-charts';
import { areaData } from 'datasource.ts';
function App() {
```

```

const primaryxAxis = {
  title: 'Year', minimum: 1900, maximum: 2000, interval: 10,
  edgeLabelPlacement: 'Shift'
};
primaryyAxis: AxisModel = { minimum: 2, maximum: 5, interval: 0.5,
title: 'Sales Amount in Millions' };
border = { width: 2 };
render();
{
  return <ChartComponent id='charts' primaryXAxis={this.primaryxAxis}
primaryYAxis={this.primaryyAxis} title='Average Sales Comparison'>
  <Inject services={[AreaSeries, Legend, Tooltip, DataLabel,
Category]}/>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={areaData} xName='x' yName='y'
name='Product A' width={2} border={this.border} opacity={0.5} type='Area'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>;
}
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, AreaSeries,
Selection}
from '@syncfusion/ej2-react-charts';
import { areaData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = {
    title: 'Year', minimum: 1900, maximum: 2000, interval: 10,
    edgeLabelPlacement: 'Shift'
  };
  const primaryyAxis: AxisModel = { minimum: 2, maximum: 5, interval: 0.5,
title: 'Sales Amount in Millions' };
  const border = { width: 2 };

  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Average Sales Comparison'>
    <Inject services={[AreaSeries, Legend, Tooltip, DataLabel, Category]}
  />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={areaData} xName='x' yName='y'
name='Product A'
        width={2} border={border}
        opacity={0.5} type='Area'>
      </SeriesDirective>

```

```

    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

See Also

- [Data label](#)
- [Tooltip](#)

Range Area in React Chart component

Range Area

To render a range area series, use series [type](#) as `RangeArea` and inject `RangeAreaSeries` module into the `services`.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, RangeAreaSeries } from
"@syncfusion/ej2-react-charts";
import { data } from "datasource.ts";
function App() {
  const marker = { visible: true, width: 10, height: 10, border: { width:
2, color: '#F8AB1D' } };
  return <ChartComponent id='charts'>
    <Inject services={[RangeAreaSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' high='high' low='low'
width={3} name='India' type='RangeArea' marker={marker}>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  </>
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
RangeAreaSeries, Selection }
from "@syncfusion/ej2-react-charts";
import { data } from "datasource.ts";
function App() {
  const marker = { visible: true, width: 10, height: 10, border: { width: 2,
color: '#F8AB1D' } };
  return <ChartComponent id='charts'>

```

```

    <Inject services={[RangeAreaSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' high='high' low='low'
width={3}
        name='India' type='RangeArea'
        marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Series customization

The following properties can be used to customize the **range area** series.

- [fill](#) – Specifies the color of the area series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes of series.
- [border](#) – Specifies the [color](#) and [width](#) of series border.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, SplineRangeAreaSeries } from
'@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
  const border = { width: 2, color: 'brown' };
  return <ChartComponent id='charts'>
    <Inject services={[RangeAreaSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' high='low' low='high'
fill='yellow' opacity='0.6' dashArray='5,5'
        type='RangeArea'
        border={border}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,

```

```

    Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
    RangeAreaSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
    const border = { width: 2, color: 'brown' };
    return <ChartComponent id='charts'>
        <Inject services={[RangeAreaSeries, Legend, Tooltip, DataLabel,
        Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' high='low' low='high'
            fill='yellow' opacity='0.6' dashArray='5,5'
            type='RangeArea'
            border={border}>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

See Also

- [Data label](#)
- [Tooltip](#)

Range step area Chart in React Chart component

Range step area

To render the range step area series, use the series [type](#) as a `RangeStepArea` and inject the `RangeStepAreaSeries` module using the `Chart.Inject(RangeStepAreaSeries)` method.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Category, RangeStepAreaSeries } from '@syncfusion/ej2-react-charts';
import { splineRangeData } from 'datasource.ts';
function App() {
    const primaryxAxis = {
        valueType: 'Category',
        edgeLabelPlacement: 'Shift',
        majorGridLines: { width: 0 }
    };
    const primaryyAxis = {
        labelFormat: '{value}°C',
        lineStyle: { width: 0 },
        minimum: 0, maximum: 40,
        majorTickLines: { width: 0 }
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis} title='Monthly Temperature Range'>

```

```

    <Inject services={[RangeStepAreaSeries, Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={splineRangeData} xName='x' high='high'
low='low' name='England' opacity={0.4} type='RangeStepArea'>
    </SeriesDirective>
      <SeriesDirective dataSource={splineRangeData} xName='x' high='high1'
low='low1' name='India' opacity={0.4} type='RangeStepArea'>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, Category, RangeStepAreaSeries }
from '@syncfusion/ej2-react-charts';
import { splineRangeData } from 'datasource.ts';
function App() {

  const primaryxAxis: AxisModel = {
    valueType: 'Category',
    edgeLabelPlacement: 'Shift',
    majorGridLines: { width: 0 }
  };
  const primaryyAxis: AxisModel = {
    labelFormat: '{value}°C',
    lineStyle: { width: 0 },
    minimum: 0, maximum: 40,
    majorTickLines: { width: 0 }
  };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Monthly Temperature Range'>
    <Inject services={[RangeStepAreaSeries, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={splineRangeData} xName='x' high='high'
low='low' name='England' opacity={0.4} type='RangeStepArea'>
    </SeriesDirective>
      <SeriesDirective dataSource={splineRangeData} xName='x' high='high1'
low='low1' name='India' opacity={0.4} type='RangeStepArea'>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

```
{% endraw %}
```

Series customization

The following properties can be used to customize the **range step area** series.

- [fill](#) – Specifies the color of the area series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes of series.
- [step](#) – Specifies the position of the step for the series.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Category, RangeStepAreaSeries } from '@syncfusion/ej2-react-charts';
import { splineRangeData } from 'datasource.ts';
function App() {
  const primaryxAxis = {
    valueType: 'Category',
    edgeLabelPlacement: 'Shift',
    majorGridLines: { width: 0 }
  };
  const primaryyAxis = {
    labelFormat: '{value}°C',
    lineStyle: { width: 0 },
    minimum: 0, maximum: 40,
    majorTickLines: { width: 0 }
  };
  const border = { color: 'yellow', width: 2};
  const border1 = { color: 'brown', width: 2};
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Monthly Temperature Range'>
    <Inject services={[RangeStepAreaSeries, Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={splineRangeData} xName='x' high='high'
low='low' name='England' opacity={0.4} type='RangeStepArea' fill='brown'
dashArray='5.5' border={border} step='Center'>
      </SeriesDirective>
      <SeriesDirective dataSource={splineRangeData} xName='x' high='high1'
low='low1' name='India' opacity={0.4} type='RangeStepArea' fill='yellow'
dashArray='5.5' border={border1} step='Center'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, Category, RangeStepAreaSeries }
from '@syncfusion/ej2-react-charts';
import { splineRangeData } from 'datasource.ts';
function App() {

  const primaryxAxis: AxisModel = {
    valueType: 'Category',
    edgeLabelPlacement: 'Shift',
    majorGridLines: { width: 0 }
  };
  const primaryyAxis: AxisModel = {
    labelFormat: '{value}°C',
    lineStyle: { width: 0 },
    minimum: 0, maximum: 40,
    majorTickLines: { width: 0 }
  };
  const border = { color: 'yellow', width: 2};
  const border1 = { color: 'brown', width: 2};
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Monthly Temperature Range'>
    <Inject services={[RangeStepAreaSeries, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={splineRangeData} xName='x' high='high'
low='low' name='England' opacity={0.4} type='RangeStepArea' fill='brown'
dashArray='5.5' border={border} step='Center'>
        </SeriesDirective>
      <SeriesDirective dataSource={splineRangeData} xName='x' high='high1'
low='low1' name='India' opacity={0.4} type='RangeStepArea' fill='yellow'
dashArray='5.5' border={border1} step='Center'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>

  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));
}{% endraw %}
```

[See also](#)

- [Data label](#)
- [Tooltip](#)

Spline Range Area in React Chart component

[Spline Range Area](#)

The Spline Range Area Chart is used to display continuous data points as a set of splines that vary between high and low values over intervals of time and across different categories.

To render a spline range area series, use series [type](#) as `SplineRangeArea` and inject `SplineRangeAreaSeries` module into the services.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Category, SplineRangeAreaSeries } from '@syncfusion/ej2-react-charts';
import { splineRangeData } from 'datasource.ts';
function App() {
  const primaryxAxis = {
    valueType: 'Category',
    edgeLabelPlacement: 'Shift',
    majorGridLines: { width: 0 }
  };
  const primaryyAxis = {
    labelFormat: '{value}°C',
    lineStyle: { width: 0 },
    minimum: 0, maximum: 40,
    majorTickLines: { width: 0 }
  };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Monthly Temperature Range'>
  <Inject services={[SplineRangeAreaSeries, Category]}/>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={splineRangeData} xName='x' high='high'
low='low' name='England' type='SplineRangeArea'>
    </SeriesDirective>
    <SeriesDirective dataSource={splineRangeData} xName='x' high='high1'
low='low1' name='India' type='SplineRangeArea'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, Category, SplineRangeAreaSeries }
from '@syncfusion/ej2-react-charts';
import { splineRangeData } from 'datasource.ts';
function App() {

  const primaryxAxis: AxisModel = {
    valueType: 'Category',
    edgeLabelPlacement: 'Shift',
    majorGridLines: { width: 0 }
  }
}
```

```

};
const primaryyAxis: AxisModel = {
  labelFormat: '{value}°C',
  lineStyle: { width: 0 },
  minimum: 0, maximum: 40,
  majorTickLines: { width: 0 }
};
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  title='Monthly Temperature Range'>
  <Inject services={[SplineRangeAreaSeries, Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={splineRangeData} xName='x' high='high'
low='low' name='England' type='SplineRangeArea'>
    </SeriesDirective>
    <SeriesDirective dataSource={splineRangeData} xName='x' high='high1'
low='low1' name='India' type='SplineRangeArea'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Series customization

The following properties can be used to customize the **spline range area** series.

- [fill](#) – Specifies the color of the area series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes of series.
- [border](#) – Specifies the [color](#) and [width](#) of series border.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Category, SplineRangeAreaSeries } from '@syncfusion/ej2-react-charts';
import { splineRangeData } from 'datasource.ts';
function App() {
  const primaryxAxis = {
    valueType: 'Category',
    edgeLabelPlacement: 'Shift',
    majorGridLines: { width: 0 }
  };
  const primaryyAxis = {
    labelFormat: '{value}°C',
    lineStyle: { width: 0 },
    minimum: 0, maximum: 40,
    majorTickLines: { width: 0 }
  };

```

```

    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Monthly Temperature Range'>
    <Inject services={[SplineRangeAreaSeries, Category]}/>
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={splineRangeData} xName='x' high='high'
low='low' name='England' opacity={0.4} type='SplineRangeArea'>
        </SeriesDirective>
        <SeriesDirective dataSource={splineRangeData} xName='x' high='high1'
low='low1' name='India' opacity={0.4} type='SplineRangeArea'>
        </SeriesDirective>
    </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, Category, SplineRangeAreaSeries }
from '@syncfusion/ej2-react-charts';
import { splineRangeData } from 'datasource.ts';
function App() {

    const primaryxAxis: AxisModel = {
        valueType: 'Category',
        edgeLabelPlacement: 'Shift',
        majorGridLines: { width: 0 }
    };
    const primaryyAxis: AxisModel = {
        labelFormat: '{value}°C',
        lineStyle: { width: 0 },
        minimum: 0, maximum: 40,
        majorTickLines: { width: 0 }
    };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Monthly Temperature Range'>
        <Inject services={[SplineRangeAreaSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={splineRangeData} xName='x' high='high'
low='low' name='England' opacity={0.7} fill='yellow' border={{width: 2,
color: 'brown'}} dashArray='5.5' type='SplineRangeArea'>
            </SeriesDirective>
            <SeriesDirective dataSource={splineRangeData} xName='x' high='high1'
low='low1' name='India' opacity={0.7} fill='brown' border={{width: 2, color:
'yellow'}} dashArray='5.5' type='SplineRangeArea'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>

```

```

    </ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

See Also

- [Data label](#)
- [Tooltip](#)

Stacked Area in React Chart component

Stacked Area

To render a stacked area series, use series [type](#) as `StackingArea` and inject `StackingAreaSeries` module into the `services`.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, StackingAreaSeries } from
'@syncfusion/ej2-react-charts';
import { stackedData } from 'datasource.ts';
function App() {
    const primaryxAxis = {
        title: 'Years', valueType: 'DateTime', intervalType: 'Years',
        majorTickLines: { width: 0 }, labelFormat: 'y', edgeLabelPlacement:
'Shift'
    };
    const primaryyAxis = {
        title: 'Spend in Billions', minimum: 0, maximum: 7, interval: 1,
        majorTickLines: { width: 0 }, labelFormat: '{value}B'
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Trend in Sales of Ethical Produce'>
        <Inject services={[StackingAreaSeries, Legend, Tooltip, DataLabel,
DateTime]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={stackedData} xName='x' yName='y'
name='Organic' type='StackingArea'>
            </SeriesDirective>
            <SeriesDirective dataSource={stackedData} xName='x' yName='y1'
name='Fair-trade' type='StackingArea'>
            </SeriesDirective>
            <SeriesDirective dataSource={stackedData} xName='x' yName='y2'
name='Veg Alternatives' type='StackingArea'>
            </SeriesDirective>
            <SeriesDirective dataSource={stackedData} xName='x' yName='y3'
name='Others' type='StackingArea'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>

```

```

    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
      Legend, DateTime, Tooltip, DataLabel, Zoom, Crosshair,
StackingAreaSeries, Selection }
from '@syncfusion/ej2-react-charts';
import { stackedData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = {
    title: 'Years', valueType: 'DateTime', intervalType: 'Years',
    majorTickLines: { width: 0 }, labelFormat: 'y', edgeLabelPlacement:
'Shift';
  const primaryyAxis: AxisModel = {
    title: 'Spend in Billions', minimum: 0, maximum: 7, interval: 1,
    majorTickLines: { width: 0 }, labelFormat: '{value}B';
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Trend in Sales of Ethical Produce'>
    <Inject services={[StackingAreaSeries, Legend, Tooltip, DataLabel,
DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={stackedData} xName='x' yName='y'
name='Organic' type='StackingArea'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackedData} xName='x' yName='y1'
name='Fair-trade' type='StackingArea'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackedData} xName='x' yName='y2'
name='Veg Alternatives' type='StackingArea'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackedData} xName='x' yName='y3'
name='Others' type='StackingArea'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Series customization

The following properties can be used to customize the **stacked area** series.

- [fill](#) – Specifies the color of the area series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes of series.
- [border](#) – Specifies the [color](#) and [width](#) of series border.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, StackingAreaSeries } from
"@syncfusion/ej2-react-charts";
import { stackedData } from "datasource.ts";
function App() {
    const primaryxAxis = {
        title: 'Years', valueType: 'DateTime', intervalType: 'Years',
        majorTickLines: { width: 0 }, labelFormat: 'y', edgeLabelPlacement:
'Shift'
    };
    const primaryyAxis = {
        title: 'Spend in Billions', minimum: 0, maximum: 7, interval: 1,
        majorTickLines: { width: 0 }, labelFormat: '{value}B'
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Trend in Sales of Ethical Produce'>
        <Inject services={[StackingAreaSeries, Legend, Tooltip, DataLabel,
DateTime]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={stackedData} xName='x' yName='y'
name='Organic' type='StackingArea' fill='grey' dashArray='5.5'
border={{width:2.5, color: 'white'}}>
            </SeriesDirective>
            <SeriesDirective dataSource={stackedData} xName='x' yName='y1'
name='Fair-trade' type='StackingArea' fill='yellow' dashArray='5.5'
border={{width:2.5, color: 'white'}}>
            </SeriesDirective>
            <SeriesDirective dataSource={stackedData} xName='x' yName='y2'
name='Veg Alternatives' type='StackingArea' fill='blue' dashArray='5.5'
border={{width:2.5, color: 'white'}}>
            </SeriesDirective>
            <SeriesDirective dataSource={stackedData} xName='x' yName='y3'
name='Others' type='StackingArea' fill='red' dashArray='5.5'
border={{width:2.5, color: 'white'}}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
      Legend, DateTime, Tooltip, DataLabel, Zoom, Crosshair,
StackingAreaSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { stackedData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = {
    title: 'Years', valueType: 'DateTime', intervalType: 'Years',
    majorTickLines: { width: 0 }, labelFormat: 'y', edgeLabelPlacement:
'Shift';
  const primaryyAxis: AxisModel = {
    title: 'Spend in Billions', minimum: 0, maximum: 7, interval: 1,
    majorTickLines: { width: 0 }, labelFormat: '{value}B';
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Trend in Sales of Ethical Produce'>
    <Inject services=[StackingAreaSeries, Legend, Tooltip, DataLabel,
DateTime]> />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={stackedData} xName='x' yName='y'
name='Organic' type='StackingArea' fill='grey' dashArray='5.5'
border={{width:2.5, color: 'white'}}>
    </SeriesDirective>
      <SeriesDirective dataSource={stackedData} xName='x' yName='y1'
name='Fair-trade' type='StackingArea' fill='yellow' dashArray='5.5'
border={{width:2.5, color: 'white'}}>
    </SeriesDirective>
      <SeriesDirective dataSource={stackedData} xName='x' yName='y2'
name='Veg Alternatives' type='StackingArea' fill='blue' dashArray='5.5'
border={{width:2.5, color: 'white'}}>
    </SeriesDirective>
      <SeriesDirective dataSource={stackedData} xName='x' yName='y3'
name='Others' type='StackingArea' fill='red' dashArray='5.5'
border={{width:2.5, color: 'white'}}>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

See Also

- [Data label](#)
- [Tooltip](#)

100% Stacked Area in React Chart component

100% Stacked Area

To render a 100% stacked area series, use series [type](#) as `StackingArea100` and inject `StackingAreaSeries` module into the `services`.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, StackingAreaSeries } from
'@syncfusion/ej2-react-charts';
import { percentData } from 'datasource.ts';
function App() {
    const primaryxAxis = {
        title: 'Years', valueType: 'DateTime', intervalType: 'Years',
        edgeLabelPlacement: 'Shift', labelFormat: 'y'
    };
    const primaryyAxis = { title: 'Temperature (%)', labelFormat:
'{{value}}%', rangePadding: 'None' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Annual Temperature Comparison'>
        <Inject services={[StackingAreaSeries, Legend, Tooltip, DataLabel,
DateTime]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={percentData} xName='x' yName='y'
name='USA' type='StackingArea100'>
            </SeriesDirective>
            <SeriesDirective dataSource={percentData} xName='x' yName='y1'
name='UK' type='StackingArea100'>
            </SeriesDirective>
            <SeriesDirective dataSource={percentData} xName='x' yName='y2'
name='Canada Alternatives' type='StackingArea100'>
            </SeriesDirective>
            <SeriesDirective dataSource={percentData} xName='x' yName='y3'
name='China' type='StackingArea100'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    Legend, DateTime, Tooltip, DataLabel, Zoom, Crosshair,
StackingAreaSeries, Selection}
from '@syncfusion/ej2-react-charts';
```



```

import { percentData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = {
    title: 'Years', valueType: 'DateTime', intervalType: 'Years',
    edgeLabelPlacement: 'Shift', labelFormat: 'y';
  }
  const primaryyAxis: AxisModel = { title: 'Temperature (%)', labelFormat:
    '{value}%', rangePadding: 'None' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Annual Temperature Comparison'>
    <Inject services=[StackingAreaSeries, Legend, Tooltip, DataLabel,
    DateTime] />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={percentData} xName='x' yName='y'
name='USA' type='StackingArea100'>
      </SeriesDirective>
      <SeriesDirective dataSource={percentData} xName='x' yName='y1'
name='UK' type='StackingArea100'>
      </SeriesDirective>
      <SeriesDirective dataSource={percentData} xName='x' yName='y2'
name='Canada Alternatives' type='StackingArea100'>
      </SeriesDirective>
      <SeriesDirective dataSource={percentData} xName='x' yName='y3'
name='China' type='StackingArea100'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Series customization

The following properties can be used to customize the **100% stacked area** series.

- [fill](#) – Specifies the color of the area series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes of series.
- [border](#) – Specifies the [color](#) and [width](#) of series border.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, StackingAreaSeries } from
'@syncfusion/ej2-react-charts';
import { percentData } from 'datasource.ts';
function App() {
  const primaryxAxis = {
    title: 'Years', valueType: 'DateTime', intervalType: 'Years',
    edgeLabelPlacement: 'Shift', labelFormat: 'y'

```

```

    };
    const primaryyAxis = { title: 'Temperature (%)', labelFormat:
    '{value}%', rangePadding: 'None' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis} title='Annual Temperature Comparison'>
      <Inject services={[StackingAreaSeries, Legend, Tooltip, DataLabel,
    DateTime]}/>
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={percentData} xName='x' yName='y'
    name='USA' type='StackingArea100' fill='red' border={{width: 2.5,
    color:'white'}} dashArray='5'>
        </SeriesDirective>
        <SeriesDirective dataSource={percentData} xName='x' yName='y1'
    name='UK' type='StackingArea100' fill='green' border={{width: 2.5,
    color:'white'}} dashArray='5'>
        </SeriesDirective>
        <SeriesDirective dataSource={percentData} xName='x' yName='y2'
    name='Canada Alternatives' type='StackingArea100' fill='orange'
    border={{width: 2.5, color:'white'}} dashArray='5'>
        </SeriesDirective>
        <SeriesDirective dataSource={percentData} xName='x' yName='y3'
    name='China' type='StackingArea100' fill='blue' border={{width: 2.5,
    color:'white'}} dashArray='5'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
    SeriesDirective, Inject,
    Legend, DateTime, Tooltip, DataLabel, Zoom, Crosshair,
    StackingAreaSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { percentData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = {
    title: 'Years', valueType: 'DateTime', intervalType: 'Years',
    edgeLabelPlacement: 'Shift', labelFormat: 'y';
  const primaryyAxis: AxisModel = { title: 'Temperature (%)', labelFormat:
  '{value}%', rangePadding: 'None' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Annual Temperature Comparison'>
    <Inject services={[StackingAreaSeries, Legend, Tooltip, DataLabel,
    DateTime]}/>
    <SeriesCollectionDirective>

```

```

    <SeriesDirective dataSource={percentData} xName='x' yName='y'
name='USA' type='StackingArea100' fill='red' border={{width: 2.5,
color:'white'}} dashArray='5'>
    </SeriesDirective>
    <SeriesDirective dataSource={percentData} xName='x' yName='y1'
name='UK' type='StackingArea100' fill='green' border={{width: 2.5,
color:'white'}} dashArray='5'>
    </SeriesDirective>
    <SeriesDirective dataSource={percentData} xName='x' yName='y2'
name='Canada Alternatives' type='StackingArea100' fill='orange'
border={{width: 2.5, color:'white'}} dashArray='5'>
    </SeriesDirective>
    <SeriesDirective dataSource={percentData} xName='x' yName='y3'
name='China' type='StackingArea100' fill='blue' border={{width: 2.5,
color:'white'}} dashArray='5'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% enddraw %}

```

See Also

- [Data label](#)
- [Tooltip](#)

Stacked step area Chart in React Chart component

Stacked step area

To render the Stacked step area series, use the series [type](#) as a `StackingStepArea` and inject the `StackingStepAreaSeries` module using the `Chart.Inject(StackingStepAreaSeries)` method.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, StackingStepAreaSeries } from '@syncfusion/ej2-react-charts';
import { stepAreaData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'Double', title: 'Overs' };
  const primaryyAxis = { title: 'Runs' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Annual Temperature Comparison'>
    <Inject services={[StackingStepAreaSeries, Legend]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={stepAreaData} xName='x' yName='y'
name='England' type='StackingStepArea'>
      </SeriesDirective>
      <SeriesDirective dataSource={stepAreaData} xName='x' yName='y1'
name='India' type='StackingStepArea'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>

```

```

    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
      Legend, StackingStepAreaSeries }
from '@syncfusion/ej2-react-charts';
import { stepAreaData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'Double', title: 'Overs' };
  const primaryyAxis: AxisModel = { title: 'Runs' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Annual Temperature Comparison'>
    <Inject services={[StackingStepAreaSeries, Legend]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={stepAreaData} xName='x' yName='y'
name='England' type='StackingStepArea'>
        </SeriesDirective>
      <SeriesDirective dataSource={stepAreaData} xName='x' yName='y1'
name='India' type='StackingStepArea' >
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  </>;
}
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Series customization

The following properties can be used to customize the **stacked step area** series.

- [fill](#) – Specifies the color of the area series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes of series.
- [step](#) – Specifies the position of the step for the series.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, StackingStepAreaSeries } from '@syncfusion/ej2-react-charts';
import { stepAreaData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'Double', title: 'Overs' };
  const primaryyAxis = { title: 'Runs' };

```

```

    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Annual Temperature Comparison'>
    <Inject services={[StackingStepAreaSeries, Legend]}/>
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={stepAreaData} xName='x' yName='y'
name='England' type='StackingStepArea' fill="red" dashArray="5"
border={{width: 2, color: 'yellow'}} opacity={0.7} step='Center'>
        </SeriesDirective>
        <SeriesDirective dataSource={stepAreaData} xName='x' yName='y1'
name='India' type='StackingStepArea' fill="green" dashArray="5"
border={{width: 2, color: 'yellow'}} opacity={0.7} step='Center'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, StackingStepAreaSeries}
from '@syncfusion/ej2-react-charts';
import { stepAreaData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Double', title: 'Overs' };
    const primaryyAxis: AxisModel = { title: 'Runs' };
    return <ChartComponent id='charts'
primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis}
title='Annual Temperature Comparison'>
    <Inject services={[StackingStepAreaSeries, Legend]} />
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={stepAreaData} xName='x' yName='y'
name='England' type='StackingStepArea' fill='red' dashArray="5"
border={{width: 2, color: 'yellow'}} opacity={0.7} step='Center'>
        </SeriesDirective>
        <SeriesDirective dataSource={stepAreaData} xName='x' yName='y1'
name='India' type='StackingStepArea' fill='green' dashArray="5"
border={{width: 2, color: 'yellow'}} opacity={0.7} step='Center'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

See also

- [Data label](#)
- [Tooltip](#)

Step area in React Chart component

Step area

To render a step area series, use series [type](#) as `StepArea` and inject `StepAreaSeries` module into the services.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, StepAreaSeries } from '@syncfusion/ej2-react-charts';
import { stepAreaData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Double', title: 'Overs' };
    const primaryyAxis = { title: 'Runs' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Annual Temperature Comparison'>
        <Inject services={[StepAreaSeries, Legend]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={stepAreaData} xName='x' yName='y'
name='England' type='StepArea'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, StepAreaSeries}
from '@syncfusion/ej2-react-charts';
import { stepAreaData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Double', title: 'Overs' };
    const primaryyAxis: AxisModel = { title: 'Runs' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Annual Temperature Comparison'>
        <Inject services={[StepAreaSeries, Legend]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={stepAreaData} xName='x' yName='y'
name='England' type='StepArea'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
    </>;
}
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Series customization

The following properties can be used to customize the **step area** series.

- [fill](#) – Specifies the color of the area series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes of series.
- [border](#) – Specifies the [color](#) and [width](#) of series border.
- [step](#) – Specifies the position of the step for the series.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, StepAreaSeries } from '@syncfusion/ej2-react-charts';
import { stepAreaData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Double', title: 'Overs' };
    const primaryyAxis = { title: 'Runs' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Annual Temperature Comparison'>
        <Inject services={[StepAreaSeries, Legend]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={stepAreaData} xName='x' yName='y'
name='England' type='StepArea' fill= 'yellow'
border={{width: 1.5, color:'brown'}} opacity={0.7} dashArray="5,5"
step='Right'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, StepAreaSeries}
from '@syncfusion/ej2-react-charts';
import { stepAreaData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Double', title: 'Overs' };
    const primaryyAxis: AxisModel = { title: 'Runs' };
    return <ChartComponent id='charts'
primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis}
title='Annual Temperature Comparison'>
        <Inject services={[StepAreaSeries, Legend]} />
        <SeriesCollectionDirective>
```

```

    <SeriesDirective dataSource={stepAreaData} xName='x' yName='y'
    type='StepArea' fill= 'yellow'
        border={{width: 1.5, color:'brown'}} opacity={0.7} dashArray="5,5"
    step='Right'>
    </SeriesDirective>
    </SeriesCollectionDirective>
    </ChartComponent>
  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));

```

See also

- [Data label](#)
- [Tooltip](#)

Step Area in React Chart component

Spline Area

To render a spline series, use series [type](#) as **Spline** and inject **SplineAreaSeries** module into the services.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
SplineAreaSeries, Legend, Category, Tooltip, DataLabel } from
'@syncfusion/ej2-react-charts';
import { splineData } from 'datasource.ts';
function App() {
  const primaryXAxis = { title: 'Month', valueType: 'Category' };
  return <ChartComponent id='charts' primaryXAxis={primaryXAxis}>
    <Inject services={[SplineAreaSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={splineData} xName='x' yName='y'
width={2} name='London' type='SplineArea' marker={{ visible: true, width:
10, height: 10 }}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";

```



```
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, SplineAreaSeries,
      Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
SplineSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { splineData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { title: 'Month', valueType: 'Category' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}>
    <Inject services={[SplineAreaSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={splineData} xName='x' yName='y'
width={2} name='London' type='SplineArea'
        marker={{ visible: true, width: 10, height: 10 }}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

Series customization

The following properties can be used to customize the **spline area** series.

- [fill](#) – Specifies the color of the area series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes of series.
- [border](#) – Specifies the [color](#) and [width](#) of series border.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
SplineAreaSeries, Legend, Category, Tooltip, DataLabel } from
 '@syncfusion/ej2-react-charts';
import { splineData } from 'datasource.ts';
function App() {
  const primaryxAxis = { title: 'Month', valueType: 'Category' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
    <Inject services={[SplineAreaSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={splineData} xName='x' yName='y' width={2}
name='London' type='SplineArea' fill='yellow' opacity='0.8' border={{width:
2, color:'brown'}} marker={{ visible: false, width: 10, height: 10 }}
dashArray='5.5'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
```

```

}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% enddraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, SplineAreaSeries,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
SplineSeries, Selection }
from '@syncfusion/ej2-react-charts';
import { splineData } from 'datasource.ts';
function App() {
  const primaryXAxis: AxisModel = { title: 'Month', valueType: 'Category' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryXAxis}>
    <Inject services={[SplineAreaSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={splineData} xName='x' yName='y'
width={2} name='London' type='SplineArea' fill='yellow' opacity='0.8'
border={{width: 2, color:'brown'}} marker={{ visible: false, width: 10,
height: 10 }} dashArray='5.5'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

See Also

- [Data label](#)
- [Tooltip](#)

Column Chart in React Chart component

Column

To render a column series, use series [type](#) as `Column` and inject `ColumnSeries` module into the services.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
import { columnData } from 'datasource.ts';

```

```
function App() {
    const primaryxAxis = { valueType: 'Category', title: 'Countries' };
    const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Olympic Medals'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={columnData} xName='country'
yName='gold' name='Gold' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
ColumnSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { columnData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
    const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Olympic Medals'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={columnData} xName='country'
yName='gold' name='Gold' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Column space and width

The [columnSpacing](#) and [columnWidth](#) properties are used to customize the space between columns.

INDEX.JSX

```
import * as React from "react";
```

```
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
import { columnData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Category', title: 'Countries' };
    const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Olympic Medals'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={columnData} xName='country'
yName='gold' name='Gold' type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={columnData} xName='country'
yName='silver' name='Silver' columnSpacing={0.5} columnWidth={0.75}
type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
ColumnSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { columnData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
    const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Olympic Medals'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={columnData} xName='country'
yName='gold' name='Gold' type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={columnData} xName='country'
yName='silver' name='Silver' columnSpacing={0.5} columnWidth={0.75}
type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
```

```

        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Grouped column

You can use the [groupName](#) property to group the data points in the column type charts. Data points with same group name are grouped together.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, ColumnSeries, DataLabel } from '@syncfusion/ej2-
react-charts';
const data1 = [
    { x: '2012', y: 104 },
    { x: '2016', y: 121 },
    { x: '2020', y: 113 },
];
const data2 = [
    { x: '2012', y: 46 },
    { x: '2016', y: 46 },
    { x: '2020', y: 39 },
];
const data3 = [
    { x: '2012', y: 65 },
    { x: '2016', y: 67 },
    { x: '2020', y: 65 },
];
const data4 = [
    { x: '2012', y: 29 },
    { x: '2016', y: 27 },
    { x: '2020', y: 22 },
];
const data5 = [
    { x: '2012', y: 91 },
    { x: '2016', y: 70 },
    { x: '2020', y: 88 },
];
const data6 = [
    { x: '2012', y: 38 },
    { x: '2016', y: 26 },
    { x: '2020', y: 38 },
];
function App() {
    return (<ChartComponent id="charts" style={{ textAlign: 'center' }}
primaryXAxis={{
        valueType: 'Category',
        interval: 1,
        majorGridLines: { width: 0 },
    }} primaryYAxis={{

```

```

        majorGridLines: { width: 0 },
        majorTickLines: { width: 0 },
        lineStyle: { width: 0 },
        labelStyle: { color: 'transparent' },
    }} chartArea={{ border: { width: 0 } }} tooltip={{ enable: true }}
title="Olympics Medal Tally">
    <Inject services=[ColumnSeries, Legend, Tooltip, Category,
DataLabel]]/>
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={data1} xName="x" yName="y" name="USA
Total" type="Column" groupName="USA" columnWidth={0.7} columnSpacing={0.1}
marker={{
            dataLabel: {
                visible: true,
                position: 'Top',
                font: { fontWeight: '600', color: '#ffffff' },
            },
        }}></SeriesDirective>
        <SeriesDirective dataSource={data2} xName="x" yName="y" name="USA
Gold" type="Column" groupName="USA" columnWidth={0.5} columnSpacing={0.1}
marker={{
            dataLabel: {
                visible: true,
                position: 'Top',
                font: { fontWeight: '600', color: '#ffffff' },
            },
        }}></SeriesDirective>
        <SeriesDirective dataSource={data3} xName="x" yName="y" name="UK
Total" type="Column" groupName="UK" columnWidth={0.7} columnSpacing={0.1}
marker={{
            dataLabel: {
                visible: true,
                position: 'Top',
                font: { fontWeight: '600', color: '#ffffff' },
            },
        }}></SeriesDirective>
        <SeriesDirective dataSource={data4} xName="x" yName="y" name="UK
Gold" type="Column" groupName="UK" columnWidth={0.5} columnSpacing={0.1}
marker={{
            dataLabel: {
                visible: true,
                position: 'Top',
                font: { fontWeight: '600', color: '#ffffff' },
            },
        }}></SeriesDirective>
        <SeriesDirective dataSource={data5} xName="x" yName="y"
name="China Total" type="Column" groupName="China" columnWidth={0.7}
columnSpacing={0.1} marker={{
            dataLabel: {
                visible: true,
                position: 'Top',
                font: { fontWeight: '600', color: '#ffffff' },
            },
        }}></SeriesDirective>
        <SeriesDirective dataSource={data6} xName="x" yName="y"
name="China Gold" type="Column" groupName="China" columnWidth={0.5}
columnSpacing={0.1} marker={{

```

```

        dataLabel: {
            visible: true,
            position: 'Top',
            font: { fontWeight: '600', color: '#ffffff' },
        },
    }}</SeriesDirective>
</SeriesCollectionDirective>
</ChartComponent>);
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
    ChartComponent,
    SeriesCollectionDirective,
    SeriesDirective,
    Inject,
    Legend,
    Category,
    Tooltip,
    ColumnSeries,
    DataLabel }
from '@syncfusion/ej2-react-charts';
const data1 = [
    { x: '2012', y: 104 },
    { x: '2016', y: 121 },
    { x: '2020', y: 113 },
];
const data2 = [
    { x: '2012', y: 46 },
    { x: '2016', y: 46 },
    { x: '2020', y: 39 },
];
const data3 = [
    { x: '2012', y: 65 },
    { x: '2016', y: 67 },
    { x: '2020', y: 65 },
];
const data4 = [
    { x: '2012', y: 29 },
    { x: '2016', y: 27 },
    { x: '2020', y: 22 },
];
const data5 = [
    { x: '2012', y: 91 },
    { x: '2016', y: 70 },
    { x: '2020', y: 88 },
];
const data6 = [

```

```

    { x: '2012', y: 38 },
    { x: '2016', y: 26 },
    { x: '2020', y: 38 },
];
function App() {
    return (
        <ChartComponent
            id="charts"
            style={{ textAlign: 'center' }}
            primaryXAxis={{
                valueType: 'Category',
                interval: 1,
                majorGridLines: { width: 0 },
            }}
            primaryYAxis={{
                majorGridLines: { width: 0 },
                majorTickLines: { width: 0 },
                lineStyle: { width: 0 },
                labelStyle: { color: 'transparent' },
            }}
            chartArea={{ border: { width: 0 } }}
            tooltip={{ enable: true }}
            title="Olympics Medal Tally"
        >
            <Inject
                services={[ColumnSeries, Legend, Tooltip, Category, DataLabel]}
            />
            <SeriesCollectionDirective>
                <SeriesDirective
                    dataSource={data1}
                    xName="x"
                    yName="y"
                    name="USA Total"
                    type="Column"
                    groupName="USA"
                    columnWidth={0.7}
                    columnSpacing={0.1}
                    marker={{
                        dataLabel: {
                            visible: true,
                            position: 'Top',
                            font: { fontWeight: '600', color: '#ffffff' },
                        },
                    }}
                ></SeriesDirective>
                <SeriesDirective
                    dataSource={data2}
                    xName="x"
                    yName="y"
                    name="USA Gold"
                    type="Column"
                    groupName="USA"
                    columnWidth={0.5}
                    columnSpacing={0.1}
                    marker={{
                        dataLabel: {
                            visible: true,

```



```

        position: 'Top',
        font: { fontWeight: '600', color: '#ffffff' },
    },
    })
</SeriesDirective>
<SeriesDirective
  dataSource={data3}
  xName="x"
  yName="y"
  name="UK Total"
  type="Column"
  groupName="UK"
  columnWidth={0.7}
  columnSpacing={0.1}
  marker={{
    dataLabel: {
      visible: true,
      position: 'Top',
      font: { fontWeight: '600', color: '#ffffff' },
    },
  }}
</SeriesDirective>
<SeriesDirective
  dataSource={data4}
  xName="x"
  yName="y"
  name="UK Gold"
  type="Column"
  groupName="UK"
  columnWidth={0.5}
  columnSpacing={0.1}
  marker={{
    dataLabel: {
      visible: true,
      position: 'Top',
      font: { fontWeight: '600', color: '#ffffff' },
    },
  }}
</SeriesDirective>
<SeriesDirective
  dataSource={data5}
  xName="x"
  yName="y"
  name="China Total"
  type="Column"
  groupName="China"
  columnWidth={0.7}
  columnSpacing={0.1}
  marker={{
    dataLabel: {
      visible: true,
      position: 'Top',
      font: { fontWeight: '600', color: '#ffffff' },
    },
  }}
</SeriesDirective>
<SeriesDirective

```

```

        dataSource={data6}
        xName="x"
        yName="y"
        name="China Gold"
        type="Column"
        groupName="China"
        columnWidth={0.5}
        columnSpacing={0.1}
        marker={{
          dataLabel: {
            visible: true,
            position: 'Top',
            font: { fontWeight: '600', color: '#ffffff' },
          },
        }}
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
);
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Cylindrical column chart

To render a cylindrical column chart, set the [columnFacet](#) property to **Cylinder** in the chart series.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
import { cylindricalData } from 'datasource.ts';
function App() {
  const primaryXAxis = { valueType: 'Category', title: 'Countries' };
  const primaryYAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
  const tooltip = { enable: true, header: "<b>${point.tooltip}</b>", format:
"Gold Medal: <b>${point.y}</b>" };
  return <ChartComponent id='charts' primaryXAxis={primaryXAxis}
primaryYAxis={primaryYAxis} tooltip={tooltip} title='Olympic Gold Medal
Counts - RIO'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category]}
  />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={cylindricalData} xName='country'
yName='gold' type='Column' columnFacet='Cylinder'
tooltipMappingName='tooltipMappingName'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;

```

```
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
  AxisModel, ChartComponent, SeriesCollectionDirective, SeriesDirective,
  Inject, TooltipSettingsModel,
  Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, ColumnSeries,
  Selection
}
  from '@syncfusion/ej2-react-charts';
import { cylindricalData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'Category', interval: 1 };
  const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 10,
  title: 'Medal Count' };
  const tooltip: TooltipSettingsModel = { enable: true, header:
  "<b>${point.tooltip}</b>", format: "Gold Medal: <b>${point.y}</b>" };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    tooltip={tooltip}
    title='Olympic Gold Medal Counts - RIO'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category]}
  />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={cylindricalData} xName='country'
yName='gold' type='Column' columnFacet='Cylinder'
tooltipMappingName='tooltipMappingName'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

Series customization

The following properties can be used to customize the **column** series.

- [fill](#) – Specifies the color of the series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes of series.
- [border](#) – Specifies the [color](#) and [width](#) of series border.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
```

```
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
import { columnData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Category', title: 'Countries' };
    const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
    const border = { color: 'brown', width: 2 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Olympic Medals'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={columnData} xName='country'
yName='gold' type='Column' fill='yellow' opacity='0.7' border={border}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
ColumnSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { columnData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
    const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
    const border = { color: 'brown', width: 2 };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Olympic Medals'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={columnData} xName='country'
yName='gold' type='Column'
                fill='yellow' border={border} opacity='0.7' dashArray='5'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

See also

- [Data label](#)
- [Tooltip](#)

Range Column Chart in React Chart component

Range Column

To render a range column series, use series [type](#) as `RangeColumn` and inject `RangeColumnSeries` module into the `services`.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, RangeColumnSeries } from
"@syncfusion/ej2-react-charts";
function App() {
    const data = [
        { x: 'Jan', low: 0.7, high: 6.1 }, { x: 'Feb', low: 1.3, high: 6.3
    }, { x: 'Mar', low: 1.9, high: 8.5 },
        { x: 'Apr', low: 3.1, high: 10.8 }, { x: 'May', low: 5.7, high:
14.40 }, { x: 'Jun', low: 8.4, high: 16.90 },
        { x: 'Jul', low: 10.6, high: 19.20 }, { x: 'Aug', low: 10.5, high:
18.9 }, { x: 'Sep', low: 8.5, high: 16.1 },
        { x: 'Oct', low: 6.0, high: 12.5 }, { x: 'Nov', low: 1.5, high: 6.9
    }, { x: 'Dec', low: 5.1, high: 12.1 }
    ];
    const data1 = [
        { x: 'Jan', low: 1.7, high: 7.1 }, { x: 'Feb', low: 1.9, high: 7.7
    }, { x: 'Mar', low: 1.2, high: 7.5 },
        { x: 'Apr', low: 2.5, high: 9.8 }, { x: 'May', low: 4.7, high: 11.4
    }, { x: 'Jun', low: 6.4, high: 14.4 },
        { x: 'Jul', low: 9.6, high: 17.2 }, { x: 'Aug', low: 10.7, high:
17.9 }, { x: 'Sep', low: 7.5, high: 15.1 },
        { x: 'Oct', low: 3.0, high: 10.5 }, { x: 'Nov', low: 1.2, high: 7.9
    }, { x: 'Dec', low: 4.1, high: 9.1 }
    ];
    const primaryxAxis = { valueType: 'Category', title: 'month' };
    const primaryyAxis = { title: 'Temperature(Celsius)' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Maximum and minimum Temperature'>
        <Inject services={[RangeColumnSeries, Legend, Tooltip,
DataLabel, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' low='low'
high='high' type='RangeColumn'>
            </SeriesDirective>
            <SeriesDirective dataSource={data1} xName='x'
low='low' high='high' type='RangeColumn'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
```

```

}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
RangeColumnSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { x: 'Jan', low: 0.7, high: 6.1 }, { x: 'Feb', low: 1.3, high:
6.3 }, { x: 'Mar', low: 1.9, high: 8.5 },
        { x: 'Apr', low: 3.1, high: 10.8 }, { x: 'May', low: 5.7, high:
14.40 }, { x: 'Jun', low: 8.4, high: 16.90 },
        { x: 'Jul', low: 10.6, high: 19.20 }, { x: 'Aug', low:
10.5, high: 18.9 }, { x: 'Sep', low: 8.5, high: 16.1 },
        { x: 'Oct', low: 6.0, high: 12.5 }, { x: 'Nov', low: 1.5, high:
6.9 }, { x: 'Dec', low: 5.1, high: 12.1 }
    ];
    const data1: any[] = [
        { x: 'Jan', low: 1.7, high: 7.1 }, { x: 'Feb', low: 1.9, high:
7.7 }, { x: 'Mar', low: 1.2, high: 7.5 },
        { x: 'Apr', low: 2.5, high: 9.8 }, { x: 'May', low: 4.7, high:
11.4 }, { x: 'Jun', low: 6.4, high: 14.4 },
        { x: 'Jul', low: 9.6, high: 17.2 }, { x: 'Aug', low: 10.7,
high: 17.9 }, { x: 'Sep', low: 7.5, high: 15.1 },
        { x: 'Oct', low: 3.0, high: 10.5 }, { x: 'Nov', low: 1.2, high:
7.9 }, { x: 'Dec', low: 4.1, high: 9.1 }
    ];
    const primaryxAxis: AxisModel = { valueType: 'Category', title: 'month' }
    ;
    const primaryyAxis: AxisModel = { title: 'Temperature(Celsius)' } ;
    return <ChartComponent id='charts'
        primaryXAxis={ primaryxAxis }
        primaryYAxis={ primaryyAxis }
        title='Maximum and minimum Temperature'>
        <Inject services={[RangeColumnSeries, Legend, Tooltip,
DataLabel, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource ={data} xName='x'
low='low' high='high' type='RangeColumn'>
            </SeriesDirective>
            <SeriesDirective dataSource ={data1} xName='x'
low='low' high='high' type='RangeColumn'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Series customization

The following properties can be used to customize the **range column** series.

- [fill](#) – Specifies the color of the series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes of series.
- [border](#) – Specifies the [color](#) and [width](#) of series border.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, RangeColumnSeries } from
"@syncfusion/ej2-react-charts";
function App() {
    const data = [
        { x: 'Jan', low: 0.7, high: 6.1 }, { x: 'Feb', low: 1.3, high: 6.3
    }, { x: 'Mar', low: 1.9, high: 8.5 },
        { x: 'Apr', low: 3.1, high: 10.8 }, { x: 'May', low: 5.7, high:
14.40 }, { x: 'Jun', low: 8.4, high: 16.90 },
        { x: 'Jul', low: 10.6, high: 19.20 }, { x: 'Aug', low: 10.5, high:
18.9 }, { x: 'Sep', low: 8.5, high: 16.1 },
        { x: 'Oct', low: 6.0, high: 12.5 }, { x: 'Nov', low: 1.5, high: 6.9
    }, { x: 'Dec', low: 5.1, high: 12.1 }
    ];
    const data1 = [
        { x: 'Jan', low: 1.7, high: 7.1 }, { x: 'Feb', low: 1.9, high: 7.7
    }, { x: 'Mar', low: 1.2, high: 7.5 },
        { x: 'Apr', low: 2.5, high: 9.8 }, { x: 'May', low: 4.7, high: 11.4
    }, { x: 'Jun', low: 6.4, high: 14.4 },
        { x: 'Jul', low: 9.6, high: 17.2 }, { x: 'Aug', low: 10.7, high:
17.9 }, { x: 'Sep', low: 7.5, high: 15.1 },
        { x: 'Oct', low: 3.0, high: 10.5 }, { x: 'Nov', low: 1.2, high: 7.9
    }, { x: 'Dec', low: 4.1, high: 9.1 }
    ];
    const primaryxAxis = { valueType: 'Category', title: 'month' };
    const primaryyAxis = { title: 'Temperature(Celsius)' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Maximum and minimum Temperature'>
        <Inject services={[RangeColumnSeries, Legend, Tooltip,
DataLabel, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' low='low'
high='high' type='RangeColumn' fill='red' border={{width:2,
color:'black'}}>
            </SeriesDirective>
            <SeriesDirective dataSource={data1} xName='x'
low='low' high='high' type='RangeColumn' fill='yellow' border={{width:2,
color:'black'}}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
}
```

```

        </ChartComponent>;
    }
    ;
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));
    {% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
        Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
RangeColumnSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { x: 'Jan', low: 0.7, high: 6.1 }, { x: 'Feb', low: 1.3, high:
6.3 }, { x: 'Mar', low: 1.9, high: 8.5 },
        { x: 'Apr', low: 3.1, high: 10.8 }, { x: 'May', low: 5.7, high:
14.40 }, { x: 'Jun', low: 8.4, high: 16.90 },
        { x: 'Jul', low: 10.6, high: 19.20 }, { x: 'Aug', low:
10.5, high: 18.9 }, { x: 'Sep', low: 8.5, high: 16.1 },
        { x: 'Oct', low: 6.0, high: 12.5 }, { x: 'Nov', low: 1.5, high:
6.9 }, { x: 'Dec', low: 5.1, high: 12.1 }
    ];
    const data1: any[] = [
        { x: 'Jan', low: 1.7, high: 7.1 }, { x: 'Feb', low: 1.9, high:
7.7 }, { x: 'Mar', low: 1.2, high: 7.5 },
        { x: 'Apr', low: 2.5, high: 9.8 }, { x: 'May', low: 4.7, high:
11.4 }, { x: 'Jun', low: 6.4, high: 14.4 },
        { x: 'Jul', low: 9.6, high: 17.2 }, { x: 'Aug', low: 10.7,
high: 17.9 }, { x: 'Sep', low: 7.5, high: 15.1 },
        { x: 'Oct', low: 3.0, high: 10.5 }, { x: 'Nov', low: 1.2, high:
7.9 }, { x: 'Dec', low: 4.1, high: 9.1 }
    ];
    const primaryxAxis: AxisModel = { valueType: 'Category', title: 'month' }
    ;
    const primaryyAxis: AxisModel = { title: 'Temperature(Celsius)' } ;
    return <ChartComponent id='charts'
        primaryXAxis={ primaryxAxis }
        primaryYAxis={ primaryyAxis }
        title='Maximum and minimum Temperature'>
        <Inject services={[RangeColumnSeries, Legend, Tooltip,
DataLabel, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource ={data} xName='x'
low='low' high='high' type='RangeColumn' fill= 'red' border={{width:2,
color:'black'}}>
                </SeriesDirective>
            <SeriesDirective dataSource ={data1} xName='x'
low='low' high='high' type='RangeColumn' fill= 'yellow' border={{width:2,
color:'black'}}>
                </SeriesDirective>

```



```

        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

See Also

- [Data label](#)
- [Tooltip](#)

Stacked Column Chart in React Chart component

Stacked column

To render a stacked column series, use series [type](#) as `StackingColumn` and inject `StackingColumnSeries` module into the `services`.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StackingColumnSeries } from
'@syncfusion/ej2-react-charts';
import { stackColumndata } from 'datasource.ts';
function App() {
    const primaryxAxis = { title: 'Years', interval: 1, valueType:
'Category' };
    const primaryyAxis = {
        title: 'Sales in Billions', minimum: 0, maximum: 700, interval: 100,
        labelFormat: '{value}B'
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Mobile Game Market by Country'>
        <Inject services={[StackingColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={stackColumndata} xName='x' yName='y'
name='UK' type='StackingColumn'>
            </SeriesDirective>
            <SeriesDirective dataSource={stackColumndata} xName='x' yName='y1'
name='Germany' type='StackingColumn'>
            </SeriesDirective>
            <SeriesDirective dataSource={stackColumndata} xName='x' yName='y2'
name='France' type='StackingColumn'>
            </SeriesDirective>
            <SeriesDirective dataSource={stackColumndata} xName='x' yName='y3'
name='Italy' type='StackingColumn'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;

```

```
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
StackingColumnSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { stackColumndata } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { title: 'Years', interval: 1, valueType:
'Category' };
  const primaryyAxis: AxisModel = {
    title: 'Sales in Billions', minimum: 0, maximum: 700, interval: 100,
    labelFormat: '{value}B'
  };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Mobile Game Market by Country'>
    <Inject services={[StackingColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={stackColumndata} xName='x' yName='y'
name='UK' type='StackingColumn'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackColumndata} xName='x' yName='y1'
name='Germany' type='StackingColumn'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackColumndata} xName='x' yName='y2'
name='France' type='StackingColumn'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackColumndata} xName='x' yName='y3'
name='Italy' type='StackingColumn'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

Cylindrical stacked column chart

To render a cylindrical stacked column chart, set the [columnFacet](#) property to **Cylinder** in the chart series.

INDEX.JSX

```
{% raw %}
```

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StackingColumnSeries } from
'@syncfusion/ej2-react-charts';
import { stackColumnndata } from 'datasource.ts';
function App() {
  const primaryxAxis = { title: 'Years', interval: 1, valueType: 'Category'
};
  const primaryyAxis = {
    title: 'Sales in Billions', minimum: 0, maximum: 700, interval: 100,
    labelFormat: '{value}B'
  };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Mobile Game Market by Country'>
    <Inject services={[StackingColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={stackColumnndata} xName='x' yName='y'
name='UK' type='StackingColumn' columnFacet='Cylinder'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackColumnndata} xName='x' yName='y1'
name='Germany' type='StackingColumn' columnFacet='Cylinder'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackColumnndata} xName='x' yName='y2'
name='France' type='StackingColumn' columnFacet='Cylinder'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackColumnndata} xName='x' yName='y3'
name='Italy' type='StackingColumn' columnFacet='Cylinder'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
  AxisModel, ChartComponent, SeriesCollectionDirective, SeriesDirective,
Inject,
  Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
StackingColumnSeries, Selection
}
  from '@syncfusion/ej2-react-charts';
import { stackColumnndata } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { title: 'Years', interval: 1, valueType:
'Category' };
  const primaryyAxis: AxisModel = {
    title: 'Sales in Billions', minimum: 0, maximum: 700, interval: 100,

```

```

    labelFormat: '{value}B'
  };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Mobile Game Market by Country'>
    <Inject services={[StackingColumnSeries, Legend, Tooltip, DataLabel,
    Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={stackColumnndata} xName='x' yName='y'
name='UK' type='StackingColumn' columnFacet='Cylinder'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackColumnndata} xName='x' yName='y1'
name='Germany' type='StackingColumn' columnFacet='Cylinder'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackColumnndata} xName='x' yName='y2'
name='France' type='StackingColumn' columnFacet='Cylinder'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackColumnndata} xName='x' yName='y3'
name='Italy' type='StackingColumn' columnFacet='Cylinder'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Series customization

The following properties can be used to customize the **stacked column** series.

- [fill](#) – Specifies the color of the series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes of series.
- [border](#) – Specifies the [color](#) and [width](#) of series border.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StackingColumnSeries } from
'@syncfusion/ej2-react-charts';
import { stackColumnndata } from 'datasource.ts';
function App() {
  const primaryxAxis = { title: 'Years', interval: 1, valueType:
'Category' };
  const primaryyAxis = {
    title: 'Sales in Billions', minimum: 0, maximum: 700, interval: 100,
    labelFormat: '{value}B'
  };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Mobile Game Market by Country'>

```

```

    <Inject services={[StackingColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={stackColumnndata} xName='x' yName='y'
name='UK' type='StackingColumn' fill='yellow' border={{width: 1, color:
'black'}} dashArray='5'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackColumnndata} xName='x' yName='y1'
name='Germany' type='StackingColumn' fill='green' border={{width: 1, color:
'black'}} dashArray='5'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackColumnndata} xName='x' yName='y2'
name='France' type='StackingColumn' fill='blue' border={{width: 1, color:
'black'}} dashArray='5'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackColumnndata} xName='x' yName='y3'
name='Italy' type='StackingColumn' fill='red' border={{width: 1, color:
'black'}} dashArray='5'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
StackingColumnSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { stackColumnndata } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { title: 'Years', interval: 1, valueType:
'Category' };
  const primaryyAxis: AxisModel = {
    title: 'Sales in Billions', minimum: 0, maximum: 700, interval: 100,
    labelFormat: '{value}B'
  };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Mobile Game Market by Country'>
    <Inject services={[StackingColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={stackColumnndata} xName='x' yName='y'
name='UK' type='StackingColumn' fill='yellow' border={{width: 1, color:
'black'}} dashArray='5'>
      </SeriesDirective>

```

```

    <SeriesDirective dataSource={stackColumnndata} xName='x' yName='y1'
name='Germany' type='StackingColumn' fill='green' border={{width: 1, color:
'black'}} dashArray='5'>
    </SeriesDirective>
    <SeriesDirective dataSource={stackColumnndata} xName='x' yName='y2'
name='France' type='StackingColumn' fill='blue' border={{width: 1, color:
'black'}} dashArray='5'>
    </SeriesDirective>
    <SeriesDirective dataSource={stackColumnndata} xName='x' yName='y3'
name='Italy' type='StackingColumn' fill='red' border={{width: 1, color:
'black'}} dashArray='5'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

See also

- [Data label](#)
- [Tooltip](#)

100% Stacked Column Chart in React Chart component

100% Stacked column

To render a 100% stacked column series, use series [type](#) as `StackingColumn100` and inject `StackingColumnSeries` module into the `services`.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StackingColumnSeries } from
'@syncfusion/ej2-react-charts';
import { columnperData } from 'datasource.ts';
function App() {
  const primaryxAxis = { title: 'Years', interval: 1, valueType:
'Category' };
  const primaryyAxis = { title: 'GDP (%) Per Annum', rangePadding: 'None',
labelFormat: '{value}%' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Gross Domestic Product Growth'>
    <Inject services={[StackingColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={columnperData} xName='x' yName='y'
name='UK' type='StackingColumn100'>
      </SeriesDirective>
      <SeriesDirective dataSource={columnperData} xName='x' yName='y1'
name='Germany' type='StackingColumn100'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>

```

```

        <SeriesDirective dataSource={columnperData} xName='x' yName='y2'
name='France' type='StackingColumn100'>
        </SeriesDirective>
        <SeriesDirective dataSource={columnperData} xName='x' yName='y3'
name='Italy' type='StackingColumn100'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
StackingColumnSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { columnperData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { title: 'Years', interval: 1, valueType:
'Category' };
    const primaryyAxis: AxisModel = { title: 'GDP (%) Per Annum',
rangePadding: 'None', labelFormat: '{value}%' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Gross Domestic Product Growth'>
        <Inject services=[<StackingColumnSeries, Legend, Tooltip, DataLabel,
Category>] />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={columnperData} xName='x' yName='y'
name='UK' type='StackingColumn100'>
            </SeriesDirective>
            <SeriesDirective dataSource={columnperData} xName='x' yName='y1'
name='Germany' type='StackingColumn100'>
            </SeriesDirective>
            <SeriesDirective dataSource={columnperData} xName='x' yName='y2'
name='France' type='StackingColumn100'>
            </SeriesDirective>
            <SeriesDirective dataSource={columnperData} xName='x' yName='y3'
name='Italy' type='StackingColumn100'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

100% Cylindrical stacked column chart

To render a 100% cylindrical stacked column chart, set the [columnFacet](#) property to **Cylinder** in the chart series.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, StackingColumnSeries } from
"@syncfusion/ej2-react-charts";
import { cylindricalData } from "datasource.ts";
function App() {
  const primaryxAxis = { title: 'Years', interval: 1, valueType: 'DateTime',
labelFormat: 'y' };
  const primaryyAxis = { title: 'GDP (%) Per Annum', rangePadding: 'None',
labelFormat: '{value}%' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Gross Domestic Product Growth'>
    <Inject services={[StackingColumnSeries, Legend, Tooltip, DataLabel,
DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={cylindricalData} xName='x' yName='y'
name='UK' type='StackingColumn100' columnFacet='Cylinder'>
      </SeriesDirective>
      <SeriesDirective dataSource={cylindricalData} xName='x' yName='y1'
name='Germany' type='StackingColumn100' columnFacet='Cylinder'>
      </SeriesDirective>
      <SeriesDirective dataSource={cylindricalData} xName='x' yName='y2'
name='France' type='StackingColumn100' columnFacet='Cylinder'>
      </SeriesDirective>
      <SeriesDirective dataSource={cylindricalData} xName='x' yName='y3'
name='Italy' type='StackingColumn100' columnFacet='Cylinder'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
  AxisModel, ChartComponent, SeriesCollectionDirective, SeriesDirective,
  Inject,
  Legend, DateTime, Tooltip, DataLabel, StackingColumnSeries
}
from "@syncfusion/ej2-react-charts";
```



```

import { cylindricalData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { title: 'Years', interval: 1, valueType:
'DateTime', labelFormat: 'y' };
  const primaryyAxis: AxisModel = { title: 'GDP (%) Per Annum',
rangePadding: 'None', labelFormat: '{value}%' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Gross Domestic Product Growth'>
    <Inject services={[StackingColumnSeries, Legend, Tooltip, DataLabel,
DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={cylindricalData} xName='x' yName='y'
name='UK' type='StackingColumn100' columnFacet='Cylinder'>
      </SeriesDirective>
      <SeriesDirective dataSource={cylindricalData} xName='x' yName='y1'
name='Germany' type='StackingColumn100' columnFacet='Cylinder'>
      </SeriesDirective>
      <SeriesDirective dataSource={cylindricalData} xName='x' yName='y2'
name='France' type='StackingColumn100' columnFacet='Cylinder'>
      </SeriesDirective>
      <SeriesDirective dataSource={cylindricalData} xName='x' yName='y3'
name='Italy' type='StackingColumn100' columnFacet='Cylinder'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Series customization

The following properties can be used to customize the 100% stacked column series.

- [fill](#) – Specifies the color of the series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes of series.
- [border](#) – Specifies the [color](#) and [width](#) of series border.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StackingColumnSeries } from
'@syncfusion/ej2-react-charts';
import { columnPerData } from 'datasource.ts';
function App() {
  const primaryxAxis = { title: 'Years', interval: 1, valueType:
'Category' };
  const primaryyAxis = { title: 'GDP (%) Per Annum', rangePadding: 'None',
labelFormat: '{value}%' };

```

```

    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Gross Domestic Product Growth'>
    <Inject services={[StackingColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={columnperData} xName='x' yName='y'
name='UK' type='StackingColumn100' fill='red' border={{width: 1, color:
'black'}}>
        </SeriesDirective>
        <SeriesDirective dataSource={columnperData} xName='x' yName='y1'
name='Germany' type='StackingColumn100' fill='yellow' border={{width: 1,
color: 'black'}}>
        </SeriesDirective>
        <SeriesDirective dataSource={columnperData} xName='x' yName='y2'
name='France' type='StackingColumn100' fill='green' border={{width: 1,
color: 'black'}}>
        </SeriesDirective>
        <SeriesDirective dataSource={columnperData} xName='x' yName='y3'
name='Italy' type='StackingColumn100' fill='blue' border={{width: 1, color:
'black'}}>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
StackingColumnSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { columnperData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { title: 'Years', interval: 1, valueType:
'Category' };
    const primaryyAxis: AxisModel = { title: 'GDP (%) Per Annum',
rangePadding: 'None', labelFormat: '{value}%' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Gross Domestic Product Growth'>
        <Inject services={[StackingColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={columnperData} xName='x' yName='y'
name='UK' type='StackingColumn100' fill='red' border={{width: 1, color:
'black'}}>
            </SeriesDirective>

```

```

        <SeriesDirective dataSource={columnperData} xName='x' yName='y1'
name='Germany' type='StackingColumn100' fill='yellow' border={{width: 1,
color: 'black'}}>
        </SeriesDirective>
        <SeriesDirective dataSource={columnperData} xName='x' yName='y2'
name='France' type='StackingColumn100' fill='blue' border={{width: 1, color:
'black'}}>
        </SeriesDirective>
        <SeriesDirective dataSource={columnperData} xName='x' yName='y3'
name='Italy' type='StackingColumn100' fill='green' border={{width: 1, color:
'black'}}>
        </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

See also

- [Data label](#)
- [Tooltip](#)

Bar Charts in React Chart component

Bar

To render a bar series, use series [type](#) as `Bar` and inject `BarSeries` module into the `services`.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, BarSeries } from '@syncfusion/ej2-
react-charts';
import { data } from 'datasource.ts';
function App() {
    const primaryxAxis = { minimum: 2005, maximum: 2012, interval: 1, title:
'Year' };
    const primaryyAxis = {
        title: 'Percentage',
        labelFormat: '{value}%'
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Unemployment rate (%)'>
        <Inject services={[BarSeries, Legend, Tooltip, DataLabel, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' name='India'
type='Bar'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}

```

```
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
      Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, BarSeries,
Selection}
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { minimum: 2005, maximum: 2012, interval:
1, title: 'Year' };
  const primaryyAxis: AxisModel = {
    title: 'Percentage',
    labelFormat: '{value}%'
  };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Unemployment rate (%)'>
    <Inject services={[BarSeries, Legend, Tooltip, DataLabel, Category]}
/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' name='India'
type='Bar'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  };
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

Bar space and width

The [columnSpacing](#) and [columnWidth](#) properties are used to customize the space between bars.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, BarSeries } from '@syncfusion/ej2-
react-charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis = { minimum: 2005, maximum: 2012, interval: 1, title:
'Year' };
  const primaryyAxis = {
```

```

        minimum: 3, maximum: 12, interval: 1, title: 'Percentage',
        labelFormat: '{value}%'
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Unemployment rate (%)'>
        <Inject services={[BarSeries, Legend, Tooltip, DataLabel, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' name='India'
type='Bar'>
                </SeriesDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y1' name='India'
type='Bar' columnSpacing={0.5} columnWidth={0.75}>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
    }
    ;
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));
    {% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
        Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, BarSeries,
Selection}
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { minimum: 2005, maximum: 2012, interval:
1, title: 'Year' };
    const primaryyAxis: AxisModel = {
        minimum: 3, maximum: 12, interval: 1, title: 'Percentage',
        labelFormat: '{value}%'
    };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Unemployment rate (%)'>
        <Inject services={[BarSeries, Legend, Tooltip, DataLabel, Category]}
/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' name='India'
type='Bar'>
                </SeriesDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y1' name='India'
type='Bar' columnSpacing={0.5} columnWidth={0.75}>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>
    };
    export default App;

```

```
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

Grouped bar

You can use the [groupName](#) property to group the data points in the Bar type charts. Data points with same group name are grouped together.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, BarSeries, DataLabel } from '@syncfusion/ej2-
react-charts';
const data1 = [
    { x: '2012', y: 104 },
    { x: '2016', y: 121 },
    { x: '2020', y: 113 },
];
const data2 = [
    { x: '2012', y: 46 },
    { x: '2016', y: 46 },
    { x: '2020', y: 39 },
];
const data3 = [
    { x: '2012', y: 65 },
    { x: '2016', y: 67 },
    { x: '2020', y: 65 },
];
const data4 = [
    { x: '2012', y: 29 },
    { x: '2016', y: 27 },
    { x: '2020', y: 22 },
];
const data5 = [
    { x: '2012', y: 91 },
    { x: '2016', y: 70 },
    { x: '2020', y: 88 },
];
const data6 = [
    { x: '2012', y: 38 },
    { x: '2016', y: 26 },
    { x: '2020', y: 38 },
];
function App() {
    return (<ChartComponent id="charts" style={{ textAlign: 'center' }}
primaryXAxis={{
        valueType: 'Category',
        interval: 1,
        majorGridLines: { width: 0 },
    }} primaryYAxis={{
        majorGridLines: { width: 0 },
        majorTickLines: { width: 0 },
        lineStyle: { width: 0 },
        labelStyle: { color: 'transparent' },

```

```

    }} chartArea={{ border: { width: 0 } }} tooltip={{ enable: true }}
    title="Olympics Medal Tally">
    <Inject services={[BarSeries, Legend, Tooltip, Category,
    DataLabel]}/>
    <SeriesCollectionDirective>
    <SeriesDirective dataSource={data1} xName="x" yName="y" name="USA
    Total" type="Bar" groupName="USA" columnWidth={0.7} columnSpacing={0.1}
    marker={{
        dataLabel: {
            visible: true,
            position: 'Top',
            font: { fontWeight: '600', color: '#ffffff' },
        },
    }}></SeriesDirective>
    <SeriesDirective dataSource={data2} xName="x" yName="y" name="USA
    Gold" type="Bar" groupName="USA" columnWidth={0.5} columnSpacing={0.1}
    marker={{
        dataLabel: {
            visible: true,
            position: 'Top',
            font: { fontWeight: '600', color: '#ffffff' },
        },
    }}></SeriesDirective>
    <SeriesDirective dataSource={data3} xName="x" yName="y" name="UK
    Total" type="Bar" groupName="UK" columnWidth={0.7} columnSpacing={0.1}
    marker={{
        dataLabel: {
            visible: true,
            position: 'Top',
            font: { fontWeight: '600', color: '#ffffff' },
        },
    }}></SeriesDirective>
    <SeriesDirective dataSource={data4} xName="x" yName="y" name="UK
    Gold" type="Bar" groupName="UK" columnWidth={0.5} columnSpacing={0.1}
    marker={{
        dataLabel: {
            visible: true,
            position: 'Top',
            font: { fontWeight: '600', color: '#ffffff' },
        },
    }}></SeriesDirective>
    <SeriesDirective dataSource={data5} xName="x" yName="y"
    name="China Total" type="Bar" groupName="China" columnWidth={0.7}
    columnSpacing={0.1} marker={{
        dataLabel: {
            visible: true,
            position: 'Top',
            font: { fontWeight: '600', color: '#ffffff' },
        },
    }}></SeriesDirective>
    <SeriesDirective dataSource={data6} xName="x" yName="y"
    name="China Gold" type="Bar" groupName="China" columnWidth={0.5}
    columnSpacing={0.1} marker={{
        dataLabel: {
            visible: true,
            position: 'Top',
            font: { fontWeight: '600', color: '#ffffff' },

```

```

    },
    >></SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>);
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
  ChartComponent,
  SeriesCollectionDirective,
  SeriesDirective,
  Inject,
  Legend,
  Category,
  Tooltip,
  BarSeries,
  DataLabel }
from '@syncfusion/ej2-react-charts';
const data1 = [
  { x: '2012', y: 104 },
  { x: '2016', y: 121 },
  { x: '2020', y: 113 },
];
const data2 = [
  { x: '2012', y: 46 },
  { x: '2016', y: 46 },
  { x: '2020', y: 39 },
];
const data3 = [
  { x: '2012', y: 65 },
  { x: '2016', y: 67 },
  { x: '2020', y: 65 },
];
const data4 = [
  { x: '2012', y: 29 },
  { x: '2016', y: 27 },
  { x: '2020', y: 22 },
];
const data5 = [
  { x: '2012', y: 91 },
  { x: '2016', y: 70 },
  { x: '2020', y: 88 },
];
const data6 = [
  { x: '2012', y: 38 },
  { x: '2016', y: 26 },
  { x: '2020', y: 38 },
];

```



```

function App() {
  return (
    <ChartComponent
      id="charts"
      style={{ textAlign: 'center' }}
      primaryXAxis={{
        valueType: 'Category',
        interval: 1,
        majorGridLines: { width: 0 },
      }}
      primaryYAxis={{
        majorGridLines: { width: 0 },
        majorTickLines: { width: 0 },
        lineStyle: { width: 0 },
        labelStyle: { color: 'transparent' },
      }}
      chartArea={{ border: { width: 0 } }}
      tooltip={{ enable: true }}
      title="Olympics Medal Tally"
    >
      <Inject
        services={[BarSeries, Legend, Tooltip, Category, DataLabel]}
      />
      <SeriesCollectionDirective>
        <SeriesDirective
          dataSource={data1}
          xName="x"
          yName="y"
          name="USA Total"
          type="Bar"
          groupName="USA"
          columnWidth={0.7}
          columnSpacing={0.1}
          marker={{
            dataLabel: {
              visible: true,
              position: 'Top',
              font: { fontWeight: '600', color: '#ffffff' },
            },
          }}
        ></SeriesDirective>
        <SeriesDirective
          dataSource={data2}
          xName="x"
          yName="y"
          name="USA Gold"
          type="Bar"
          groupName="USA"
          columnWidth={0.5}
          columnSpacing={0.1}
          marker={{
            dataLabel: {
              visible: true,
              position: 'Top',
              font: { fontWeight: '600', color: '#ffffff' },
            },
          }}
        ></SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  )
}

```

```

></SeriesDirective>
<SeriesDirective
  dataSource={data3}
  xName="x"
  yName="y"
  name="UK Total"
  type="Bar"
  groupName="UK"
  columnWidth={0.7}
  columnSpacing={0.1}
  marker={{
    dataLabel: {
      visible: true,
      position: 'Top',
      font: { fontWeight: '600', color: '#ffffff' },
    },
  }}
></SeriesDirective>
<SeriesDirective
  dataSource={data4}
  xName="x"
  yName="y"
  name="UK Gold"
  type="Bar"
  groupName="UK"
  columnWidth={0.5}
  columnSpacing={0.1}
  marker={{
    dataLabel: {
      visible: true,
      position: 'Top',
      font: { fontWeight: '600', color: '#ffffff' },
    },
  }}
></SeriesDirective>
<SeriesDirective
  dataSource={data5}
  xName="x"
  yName="y"
  name="China Total"
  type="Bar"
  groupName="China"
  columnWidth={0.7}
  columnSpacing={0.1}
  marker={{
    dataLabel: {
      visible: true,
      position: 'Top',
      font: { fontWeight: '600', color: '#ffffff' },
    },
  }}
></SeriesDirective>
<SeriesDirective
  dataSource={data6}
  xName="x"
  yName="y"
  name="China Gold"

```

```

        type="Bar"
        groupName="China"
        columnWidth={0.5}
        columnSpacing={0.1}
        marker={{
            dataLabel: {
                visible: true,
                position: 'Top',
                font: { fontWeight: '600', color: '#ffffff' },
            },
        }}
    </SeriesDirective>
</SeriesCollectionDirective>
</ChartComponent>
);
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Cylindrical bar chart

To render a cylindrical bar chart, set the [columnFacet](#) property to `Cylinder` in the chart series.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, BarSeries } from '@syncfusion/ej2-
react-charts';
import { cylindricalData } from 'datasource.ts';
function App() {
    const primaryxAxis = { minimum: 2005, maximum: 2012, interval: 1 };
    const primaryyAxis = {
        minimum: 3,
        maximum: 12,
        interval: 1,
        title: 'Percentage'
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Unemployment rate in percentage'>
        <Inject services={[BarSeries, Legend, Tooltip, DataLabel, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={cylindricalData} xName='x' yName='y'
name='India' type='Bar' columnFacet='Cylinder'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, BarSeries,
Selection}
from '@syncfusion/ej2-react-charts';
import { cylindricalData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { minimum: 2005, maximum: 2012, interval:
1 };
  const primaryyAxis: AxisModel = {
    minimum: 3,
    maximum: 12,
    interval: 1,
    title: 'Percentage'
  };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Unemployment rate in percentage'>
    <Inject services={[BarSeries, Legend, Tooltip, DataLabel, Category]}
/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={cylindricalData} xName='x' yName='y'
name='India' type='Bar' columnFacet='Cylinder'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

Series customization

The following properties can be used to customize the **bar** series.

- [fill](#) – Specifies the color of the series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes of series.
- [border](#) – Specifies the [color](#) and [width](#) of series border.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, BarSeries } from '@syncfusion/ej2-
react-charts';
import { customData } from 'datasource.ts';
function App() {
```

```

    const primaryxAxis = { minimum: 2005, maximum: 2012, interval: 1, title:
'Year' };
    const primaryyAxis = {
      minimum: 3, maximum: 12, interval: 1, title: 'Percentage',
      labelFormat: '{value}%'
    };
    const border = { color: 'brown', width: 2 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Unemployment rate (%)'>
      <Inject services={[BarSeries, Legend, Tooltip, DataLabel, Category]}/>
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={customData} xName='x' yName='y'
type='Bar' pointColorMapping= 'point' dashArray='2' border={border}>
          </SeriesDirective>
        </SeriesCollectionDirective>
      </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
      Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, BarSeries,
Selection}
from '@syncfusion/ej2-react-charts';
import { customData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { minimum: 2005, maximum: 2012, interval:
1, title: 'Year' };
  const primaryyAxis: AxisModel = {
    minimum: 3, maximum: 12, interval: 1, title: 'Percentage',
    labelFormat: '{value}%'
  };
  const border = { color: 'brown', width: 2 };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Unemployment rate (%)'>
    <Inject services={[BarSeries, Legend, Tooltip, DataLabel, Category]}
  />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={customData} xName='x' yName='y'
type='Bar'
      border={border} pointColorMapping= 'point' dashArray='2'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  };
export default App;

```

```
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

See also

- [Data label](#)
- [Tooltip](#)

Stacked Bar Charts in React Chart component

Stacked bar

To render a stacked bar series, use series [type](#) as `StackingBar` and inject `StackingBarSeries` module into the `services`.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StackingBarSeries } from
"@syncfusion/ej2-react-charts";
import { stackBarData } from "datasource.ts";
function App() {
    const primaryxAxis = { valueType: 'Category', title: 'Months' };
    const primaryyAxis = {
        title: 'Percentage (%)', minimum: -20, maximum: 100,
        edgeLabelPlacement: 'Shift', labelFormat: '{value}%'
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Sales Comparison'>
        <Inject services={[StackingBarSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={stackBarData} xName='x' yName='y'
name='Apple' type='StackingBar'>
            </SeriesDirective>
            <SeriesDirective dataSource={stackBarData} xName='x' yName='y1'
name='orange' type='StackingBar'>
            </SeriesDirective>
            <SeriesDirective dataSource={stackBarData} xName='x' yName='y2'
name='Wastage' type='StackingBar'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
```

```
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
      Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
StackingBarSeries, Selection }
from '@syncfusion/ej2-react-charts';
import { stackBarData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'Category', title: 'Months'
};
  const primaryyAxis: AxisModel = {
    title: 'Percentage (%)', minimum: -20, maximum: 100,
    edgeLabelPlacement: 'Shift', labelFormat: '{value}%'
  };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Sales Comparison'>
    <Inject services={[StackingBarSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={stackBarData} xName='x' yName='y'
name='Apple' type='StackingBar'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackBarData} xName='x' yName='y1'
name='orange' type='StackingBar'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackBarData} xName='x' yName='y2'
name='Wastage' type='StackingBar'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

Stacking group

You can use the [stackingGroup](#) property to group the stacked bar and 100% stacked bar. Columns with same group name are stacked on top of each other.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StackingBarSeries } from
 '@syncfusion/ej2-react-charts';
import { groupData } from 'datasource.ts';
function App() {
  const primaryxAxis = { title: 'Year', minimum: 2006, maximum: 2015,
interval: 1 };
  const primaryyAxis = { title: 'Sales Percentage(%)' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Sales by year'>
```

```

    <Inject services={[StackingBarSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={groupData} xName='x' yName='y'
name='John' type='StackingBar' stackingGroup='JohnandAndrew'>
    </SeriesDirective>
      <SeriesDirective dataSource={groupData} xName='x' yName='y1'
name='Andrew' type='StackingBar' stackingGroup='JohnandAndrew'>
    </SeriesDirective>
      <SeriesDirective dataSource={groupData} xName='x' yName='y2'
name='Thomas' type='StackingBar' stackingGroup='ThomasandMichael'>
    </SeriesDirective>
      <SeriesDirective dataSource={groupData} xName='x' yName='y3'
name='Michael' type='StackingBar' stackingGroup='ThomasandMichael'>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
      Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
StackingBarSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { groupData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { title: 'Year', minimum: 2006, maximum:
2015, interval: 1 };
  const primaryyAxis: AxisModel = { title: 'Sales Percentage(%)' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Sales by year'>
    <Inject services={[StackingBarSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={groupData} xName='x' yName='y'
name='John' type='StackingBar'
      stackingGroup='JohnandAndrew'>
    </SeriesDirective>
      <SeriesDirective dataSource={groupData} xName='x' yName='y1'
name='Andrew' type='StackingBar'
      stackingGroup='JohnandAndrew'>
    </SeriesDirective>
      <SeriesDirective dataSource={groupData} xName='x' yName='y2'
name='Thomas' type='StackingBar'
      stackingGroup='ThomasandMichael'>
    </SeriesDirective>

```



```

        <SeriesDirective dataSource={groupData} xName='x' yName='y3'
name='Michael' type='StackingBar'
        stackingGroup='ThomasandMichael'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Cylindrical stacked bar chart

To render a cylindrical stacked bar chart, set the [columnFacet](#) property to **Cylinder** in the chart series.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StackingBarSeries } from
'@syncfusion/ej2-react-charts';
import { cylindricalData } from 'datasource.ts';
function App() {
    return <ChartComponent id='charts'>
        <Inject services={[StackingBarSeries, Legend, Tooltip, DataLabel,
Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={cylindricalData} xName='x' yName='y'
type='StackingBar' columnFacet='Cylinder'>
            </SeriesDirective>
            <SeriesDirective dataSource={cylindricalData} xName='x' yName='y1'
type='StackingBar' columnFacet='Cylinder'>
            </SeriesDirective>
            <SeriesDirective dataSource={cylindricalData} xName='x' yName='y2'
type='StackingBar' columnFacet='Cylinder'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
    AxisModel, ChartComponent, SeriesCollectionDirective, SeriesDirective,
    Inject,
    Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, StackingBarSeries,
    Selection
}
    from '@syncfusion/ej2-react-charts';

```

```
import { cylindricalData } from 'datasource.ts';
function App() {
  return <ChartComponent id='charts'>
    <Inject services={[StackingBarSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={cylindricalData} xName='x' yName='y'
type='StackingBar' columnFacet='Cylinder'>
      </SeriesDirective>
      <SeriesDirective dataSource={cylindricalData} xName='x' yName='y1'
type='StackingBar' columnFacet='Cylinder'>
      </SeriesDirective>
      <SeriesDirective dataSource={cylindricalData} xName='x' yName='y2'
type='StackingBar' columnFacet='Cylinder'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

Series customization

The following properties can be used to customize the **stacked bar** series.

- [fill](#) – Specifies the color of the series.
- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes of series.
- [border](#) – Specifies the [color](#) and [width](#) of series border.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StackingBarSeries } from
'@syncfusion/ej2-react-charts';
import { stackBarData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'Category', title: 'Months' };
  const primaryyAxis = {
    title: 'Percentage (%)', minimum: -20, maximum: 100,
    edgeLabelPlacement: 'Shift', labelFormat: '{value}%'
  };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Sales Comparison'>
    <Inject services={[StackingBarSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={stackBarData} xName='x' yName='y'
dashArray='5' type='StackingBar' fill='brown' opacity='0.8' border= {{width:
.5, color: 'yellow'}}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

```

        <SeriesDirective dataSource={stackBarData} xName='x' yName='y1'
dashArray='5' type='StackingBar' fill='grey' opacity='0.8' border= {{width:
.5, color: 'red'}}>
        </SeriesDirective>
        <SeriesDirective dataSource={stackBarData} xName='x' yName='y2'
dashArray='5' type='StackingBar' fill='yellow' opacity='0.8' border=
{{width: .5, color: 'brown'}}>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
        Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
StackingBarSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { stackBarData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Category', title: 'Months'
};
    const primaryyAxis: AxisModel = {
        title: 'Percentage (%)', minimum: -20, maximum: 100,
        edgeLabelPlacement: 'Shift', labelFormat: '{value}%'
    };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Sales Comparison'>
        <Inject services={[StackingBarSeries, Legend, Tooltip, DataLabel,
Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={stackBarData} xName='x' yName='y'
dashArray='5' type='StackingBar' fill='brown' opacity='0.8' border= {{width:
.5, color: 'yellow'}}>
            </SeriesDirective>
            <SeriesDirective dataSource={stackBarData} xName='x' yName='y1'
dashArray='5' type='StackingBar' fill='grey' opacity='0.8' border= {{width:
.5, color: 'red'}}>
            </SeriesDirective>
            <SeriesDirective dataSource={stackBarData} xName='x' yName='y2'
dashArray='5' type='StackingBar' fill='yellow' opacity='0.8' border=
{{width: .5, color: 'brown'}}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};

```

```
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

See also

- [Data label](#)
- [Tooltip](#)

100% Stacked Bar Charts in React Chart component

100% Stacked bar

To render a 100% stacked bar series, use series [type](#) as `StackingBar100` and inject `StackingBarSeries` module into the `services`.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StackingBarSeries } from
"@syncfusion/ej2-react-charts";
import { stackBarData } from "datasource.ts";
function App() {
    const primaryxAxis = { valueType: 'Category', title: 'Months' };
    const primaryyAxis = {
        title: 'Percentage (%)', minimum: -20, maximum: 100,
        edgeLabelPlacement: 'Shift', labelFormat: '{value}%'
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Sales Comparison'>
        <Inject services={[StackingBarSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={stackBarData} xName='x' yName='y'
name='Apple' type='StackingBar100'>
            </SeriesDirective>
            <SeriesDirective dataSource={stackBarData} xName='x' yName='y1'
name='orange' type='StackingBar100'>
            </SeriesDirective>
            <SeriesDirective dataSource={stackBarData} xName='x' yName='y2'
name='Wastage' type='StackingBar100'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
```

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
      Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
StackingBarSeries, Selection }
from '@syncfusion/ej2-react-charts';
import { stackBarData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'Category', title: 'Months'
};
  const primaryyAxis: AxisModel = {
    title: 'Percentage (%)', minimum: -20, maximum: 100,
    edgeLabelPlacement: 'Shift', labelFormat: '{value}%'
  };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Sales Comparison'>
    <Inject services={[StackingBarSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={stackBarData} xName='x' yName='y'
name='Apple' type='StackingBar100'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackBarData} xName='x' yName='y1'
name='orange' type='StackingBar100'>
      </SeriesDirective>
      <SeriesDirective dataSource={stackBarData} xName='x' yName='y2'
name='Wastage' type='StackingBar100'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

100% Cylindrical stacked bar chart

To render a 100% cylindrical stacked bar chart, set the [columnFacet](#) property to **Cylinder** in the chart series.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StackingBarSeries } from
 '@syncfusion/ej2-react-charts';
import { cylindricalData } from 'datasource.ts';
function App() {
  return <ChartComponent id='charts'>
    <Inject services={[StackingBarSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>

```

```

    <SeriesDirective dataSource={cylindricalData} xName='x' yName='y'
type='StackingBar100' columnFacet='Cylinder'>
    </SeriesDirective>
    <SeriesDirective dataSource={cylindricalData} xName='x' yName='y1'
type='StackingBar100' columnFacet='Cylinder'>
    </SeriesDirective>
    <SeriesDirective dataSource={cylindricalData} xName='x' yName='y2'
type='StackingBar100' columnFacet='Cylinder'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
  ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
  Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, StackingBarSeries,
  Selection
}
  from '@syncfusion/ej2-react-charts';
import { cylindricalData } from 'datasource.ts';
function App() {
  return <ChartComponent id='charts'>
    <Inject services={[StackingBarSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={cylindricalData} xName='x' yName='y'
type='StackingBar100' columnFacet='Cylinder'>
      </SeriesDirective>
      <SeriesDirective dataSource={cylindricalData} xName='x' yName='y1'
type='StackingBar100' columnFacet='Cylinder'>
      </SeriesDirective>
      <SeriesDirective dataSource={cylindricalData} xName='x' yName='y2'
type='StackingBar100' columnFacet='Cylinder'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Series customization

The following properties can be used to customize the **100% stacked bar** series.

- [fill](#) – Specifies the color of the series.

- [opacity](#) – Specifies the opacity of [fill](#).
- [dashArray](#) – Specifies the dashes for series.
- [border](#) – Specifies the [color](#) and [width](#) of series border.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StackingBarSeries } from
'@syncfusion/ej2-react-charts';
import { stackBarData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Category', title: 'Months' };
    const primaryyAxis = {
        title: 'Percentage (%)', minimum: -20, maximum: 100,
        edgeLabelPlacement: 'Shift', labelFormat: '{value}%'
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Sales Comparison'>
        <Inject services={[StackingBarSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={stackBarData} xName='x' yName='y'
name='Apple' type='StackingBar100' fill='red' border={{width: 2, color:
'black'}}>
            </SeriesDirective>
            <SeriesDirective dataSource={stackBarData} xName='x' yName='y1'
name='orange' type='StackingBar100' fill='blue' border={{width: 2, color:
'black'}}>
            </SeriesDirective>
            <SeriesDirective dataSource={stackBarData} xName='x' yName='y2'
name='Wastage' type='StackingBar100' fill='yellow' border={{width: 2,
color: 'black'}}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
StackingBarSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { stackBarData } from 'datasource.ts';
function App() {
```

```

const primaryxAxis: AxisModel = { valueType: 'Category', title: 'Months'
};
const primaryyAxis: AxisModel = {
  title: 'Percentage (%)', minimum: -20, maximum: 100,
  edgeLabelPlacement: 'Shift', labelFormat: '{value}%'
};
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  title='Sales Comparison'>
  <Inject services={[StackingBarSeries, Legend, Tooltip, DataLabel,
Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={stackBarData} xName='x' yName='y'
name='Apple' type='StackingBar100' fill='red' border= {{width: 2, color:
'black'}}>
    </SeriesDirective>
    <SeriesDirective dataSource={stackBarData} xName='x' yName='y1'
name='orange' type='StackingBar100' fill='yellow' border= {{width: 2,
color: 'black'}}>
    </SeriesDirective>
    <SeriesDirective dataSource={stackBarData} xName='x' yName='y2'
name='Wastage' type='StackingBar100' fill='green' border= {{width: 2,
color: 'black'}}>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

See also

- [Data label](#)
- [Tooltip](#)

Bubble in React Chart component

Scatter

To render a scatter series, use series [type](#) as **Scatter** and inject **ScatterSeries** module into the **services**.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ScatterSeries } from '@syncfusion/ej2-
react-charts';
import { scatterData } from 'datasource.ts';
function App() {
  const primaryxAxis = {
    title: 'Height (cm)', minimum: 130, maximum: 180,
    edgeLabelPlacement: 'Shift', labelFormat: '{value}cm'
  };

```



```

    const primaryyAxis = {
      title: 'Weight (kg)', minimum: 30, maximum: 100,
      labelFormat: '{value}kg', rangePadding: 'None'
    };
    const marker = { width: 15, height: 15 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Height Vs Weight'>
      <Inject services={[ScatterSeries, Legend, Tooltip, DataLabel,
Category]}/>
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={scatterData} xName='height'
yName='male' name='Male' type='Scatter' marker={marker}>
          </SeriesDirective>
        <SeriesDirective dataSource={scatterData} xName='height'
yName='female' name='Female' type='Scatter' marker={marker}>
          </SeriesDirective>
        </SeriesCollectionDirective>
      </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
      Legend, Category, Tooltip, DataLabel, ScatterSeries, Marker}
from '@syncfusion/ej2-react-charts';
import { scatterData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = {
    title: 'Height (cm)', minimum: 130, maximum: 180,
    edgeLabelPlacement: 'Shift', labelFormat: '{value}cm'
  };
  const primaryyAxis: AxisModel = {
    title: 'Weight (kg)', minimum: 30, maximum: 100,
    labelFormat: '{value}kg', rangePadding: 'None'
  };
  const marker = { width: 15, height: 15 };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Height Vs Weight'>
    <Inject services={[ScatterSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={scatterData} xName='height'
yName='male' name='Male' type='Scatter'
        marker={marker}>
        </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

```

        <SeriesDirective dataSource={scatterData} xName='height'
yName='female' name='Female' type='Scatter'
        marker={marker}>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Series customization

The following properties can be used to customize the **scatter** series.

- [fill](#) – Specifies the color of the series.
- [opacity](#) – Specifies the opacity of [Fill](#).
- [shape](#) - Specifies the shape of the scatter series.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ScatterSeries } from '@syncfusion/ej2-
react-charts';
import { scatterData } from 'datasource.ts';
function App() {
    const primaryxAxis = {
        title: 'Height (cm)', minimum: 130, maximum: 180,
        edgeLabelPlacement: 'Shift', labelFormat: '{value}cm'
    };
    const primaryyAxis = {
        title: 'Weight (kg)', minimum: 30, maximum: 100,
        labelFormat: '{value}kg', rangePadding: 'None'
    };
    const marker1 = { width: 10, height: 10, shape: 'Triangle' };
    const marker = { width: 12, height: 12, shape: 'Rectangle' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Height Vs Weight'>
        <Inject services={[ScatterSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={scatterData} xName='height' yName='male'
type='Scatter' fill='red' opacity='0.7'
            marker={marker1}>
            </SeriesDirective>
            <SeriesDirective dataSource={scatterData} xName='height'
yName='female' type='Scatter' fill='yellow' opacity='0.7'
            marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}

```

```
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
      Legend, Category, Tooltip, DataLabel, ScatterSeries, Marker}
from '@syncfusion/ej2-react-charts';
import { scatterData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = {
    title: 'Height (cm)', minimum: 130, maximum: 180,
    edgeLabelPlacement: 'Shift', labelFormat: '{value}cm'
  };
  const primaryyAxis: AxisModel = {
    title: 'Weight (kg)', minimum: 30, maximum: 100,
    labelFormat: '{value}kg', rangePadding: 'None'
  };
  const marker1 = { width: 10, height: 10, shape: 'Triangle' };
  const marker = { width: 12, height: 12, shape: 'Rectangle' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Height Vs Weight'>
    <Inject services={[ScatterSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={scatterData} xName='height'
yName='male' type='Scatter' fill='red' opacity='0.7'
        marker={marker1}>
      </SeriesDirective>
      <SeriesDirective dataSource={scatterData} xName='height'
yName='female' type='Scatter' fill='yellow' opacity='0.7'
        marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

See Also

- [Data label](#)
- [Tooltip](#)

Scatter in React Chart component

Bubble

To render a bubble series, use series [type](#) as `Bubble` and inject `BubbleSeries` module into the `services`.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
BubbleSeries } from '@syncfusion/ej2-react-charts';
function App() {
  const data = [
    { x: 92.2, y: 7.8, size: 1.347, text: 'China' },
    { x: 74, y: 6.5, size: 1.241, text: 'India' },
    { x: 90.4, y: 6.0, size: 0.238, text: 'Indonesia' },
    { x: 99.4, y: 2.2, size: 0.312, text: 'US' },
    { x: 88.6, y: 1.3, size: 0.197, text: 'Brazil' },
    { x: 99, y: 0.7, size: 0.0818, text: 'Germany' },
    { x: 72, y: 2.0, size: 0.0826, text: 'Egypt' },
    { x: 99.6, y: 3.4, size: 0.143, text: 'Russia' },
    { x: 99, y: 0.2, size: 0.128, text: 'Japan' },
    { x: 86.1, y: 4.0, size: 0.115, text: 'Mexico' },
    { x: 92.6, y: 6.6, size: 0.096, text: 'Philippines' },
    { x: 61.3, y: 14.5, size: 0.162, text: 'Nigeria' }
  ];
  const primaryxAxis = { title: 'Literacy Rate', minimum: 60, maximum:
100, interval: 5 };
  const primaryyAxis = { title: 'GDP growth rate', minimum: -2, maximum:
16, interval: 2 };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='GDP vs Literacy Rate'>
    <Inject services={[BubbleSeries]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y'
size='size' type='Bubble' name='pound'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
BubbleSeries}
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { x: 92.2, y: 7.8, size: 1.347, text: 'China' },
    { x: 74, y: 6.5, size: 1.241, text: 'India' },
    { x: 90.4, y: 6.0, size: 0.238, text: 'Indonesia' },
```

```

    { x: 99.4, y: 2.2, size: 0.312, text: 'US' },
    { x: 88.6, y: 1.3, size: 0.197, text: 'Brazil' },
    { x: 99, y: 0.7, size: 0.0818, text: 'Germany' },
    { x: 72, y: 2.0, size: 0.0826, text: 'Egypt' },
    { x: 99.6, y: 3.4, size: 0.143, text: 'Russia' },
    { x: 99, y: 0.2, size: 0.128, text: 'Japan' },
    { x: 86.1, y: 4.0, size: 0.115, text: 'Mexico' },
    { x: 92.6, y: 6.6, size: 0.096, text: 'Philippines' },
    { x: 61.3, y: 14.5, size: 0.162, text: 'Nigeria' }
  ];
  const primaryxAxis: AxisModel= { title: 'Literacy Rate', minimum: 60,
maximum: 100, interval: 5 } ;
  const primaryyAxis: AxisModel= { title: 'GDP growth rate', minimum: -2,
maximum: 16, interval: 2 } ;
  return <ChartComponent id='charts'
    primaryXAxis={ primaryxAxis }
    primaryYAxis={ primaryyAxis }
    title='GDP vs Literacy Rate'>
    <Inject services={[BubbleSeries]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource ={data} xName='x' yName='y'
size='size' type='Bubble' name='pound'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Size mapping

size property can be used to map the size value specified in data source.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
BubbleSeries } from '@syncfusion/ej2-react-charts';
import { bubbleData } from 'datasource.ts';
function App() {
  return <ChartComponent id='charts'>
    <Inject services={[BubbleSeries]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={bubbleData} xName='x' yName='y'
size='size' type='Bubble'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";

```

```
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
  BubbleSeries }
from '@syncfusion/ej2-react-charts';
import { bubbleData } from 'datasource.ts';
function App() {
  return <ChartComponent id='charts'>
    <Inject services={[BubbleSeries]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={bubbleData} xName='x' yName='y'
size='size' type='Bubble'>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Series customization

The following properties can be used to customize the **bubble** series.

- [fill](#) – Specifies the color of the series.
- [opacity](#) – Specifies the opacity of [fill](#).

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
  BubbleSeries } from '@syncfusion/ej2-react-charts';
function App() {
  const data = [
    { x: 92.2, y: 7.8, size: 1.347, text: 'China' },
    { x: 74, y: 6.5, size: 1.241, text: 'India' },
    { x: 90.4, y: 6.0, size: 0.238, text: 'Indonesia' },
    { x: 99.4, y: 2.2, size: 0.312, text: 'US' },
    { x: 88.6, y: 1.3, size: 0.197, text: 'Brazil' },
    { x: 99, y: 0.7, size: 0.0818, text: 'Germany' },
    { x: 72, y: 2.0, size: 0.0826, text: 'Egypt' },
    { x: 99.6, y: 3.4, size: 0.143, text: 'Russia' },
    { x: 99, y: 0.2, size: 0.128, text: 'Japan' },
    { x: 86.1, y: 4.0, size: 0.115, text: 'Mexico' },
    { x: 92.6, y: 6.6, size: 0.096, text: 'Philippines' },
    { x: 61.3, y: 14.5, size: 0.162, text: 'Nigeria' }
  ];
  const primaryxAxis = { title: 'Literacy Rate', minimum: 60, maximum:
100, interval: 5 };
  const primaryyAxis = { title: 'GDP growth rate', minimum: -2, maximum:
16, interval: 2 };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='GDP vs Literacy Rate'>
    <Inject services={[BubbleSeries]} />
    <SeriesCollectionDirective>
```

```

        <SeriesDirective dataSource={data} xName='x' yName='y'
size='size' type='Bubble' name='pound' fill='blue' border={{width:2, color:
'black'}}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    BubbleSeries }
from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { x: 92.2, y: 7.8, size: 1.347, text: 'China' },
        { x: 74, y: 6.5, size: 1.241, text: 'India' },
        { x: 90.4, y: 6.0, size: 0.238, text: 'Indonesia' },
        { x: 99.4, y: 2.2, size: 0.312, text: 'US' },
        { x: 88.6, y: 1.3, size: 0.197, text: 'Brazil' },
        { x: 99, y: 0.7, size: 0.0818, text: 'Germany' },
        { x: 72, y: 2.0, size: 0.0826, text: 'Egypt' },
        { x: 99.6, y: 3.4, size: 0.143, text: 'Russia' },
        { x: 99, y: 0.2, size: 0.128, text: 'Japan' },
        { x: 86.1, y: 4.0, size: 0.115, text: 'Mexico' },
        { x: 92.6, y: 6.6, size: 0.096, text: 'Philippines' },
        { x: 61.3, y: 14.5, size: 0.162, text: 'Nigeria' }
    ];
    const primaryxAxis: AxisModel= { title: 'Literacy Rate', minimum: 60,
maximum: 100, interval: 5 } ;
    const primaryyAxis: AxisModel= { title: 'GDP growth rate', minimum: -2,
maximum: 16, interval: 2 } ;
    return <ChartComponent id='charts'
        primaryXAxis={ primaryxAxis }
        primaryYAxis={ primaryyAxis }
        title='GDP vs Literacy Rate'>
        <Inject services={[BubbleSeries]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource ={data} xName='x' yName='y'
size='size' type='Bubble' name='pound' fill='blue' border={{width:2, color:
'black'}}>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>
    };
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));
}

```

See Also

- [Data label](#)
- [Tooltip](#)

Polar Charts in React Chart component

Polar Chart

To render a polar series, use series [type](#) as **Polar** and inject [Link to the Video](#) module into services

To get start quickly with React Polar and Radar Charts, you can check on this video:

Draw Types

Polar drawType property is used to change the series plotting type to line, column, area, range column,spline, scatter, stacking area and stacking column. The default value of drawType is **Line**.

Line

To render a line draw type, use series [drawType](#) as **Line** and inject **LineSeries** into services.

[isClosed](#) property specifies whether to join start and end point of a line series used in polar chart to form a closed path. Default value of isClosed is true.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
PolarSeries, LineSeries } from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
    const primaryxAxis = { title: 'Month' };
    const primaryyAxis = { minimum: 20, maximum: 40, interval: 5, title:
'Efficiency', labelFormat: '{value}%' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Efficiency of oil-fired power
production'>
        <Inject services={[PolarSeries, LineSeries]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' type='Polar'
name='Department' drawType='Line'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    PolarSeries, LineSeries}
```



```

from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { title: 'Month' };
  const primaryyAxis: AxisModel = { minimum: 20, maximum: 40, interval: 5,
title: 'Efficiency', labelFormat: '{value}%' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Efficiency of oil-fired power production'>
    <Inject services={[PolarSeries, LineSeries]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' type='Polar'
name='Department' drawType='Line'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  };
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Spline

To render a spline draw type, use series [drawType](#) as **Spline** and inject **SplineSeries** into services.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
PolarSeries, Category, SplineSeries } from '@syncfusion/ej2-react-charts';
import { splineData } from 'datasource.ts';
function App() {
  const primaryxAxis = { title: 'Month', valueType: 'Category' };
  const primaryyAxis = { minimum: -5, maximum: 35, interval: 10, title:
'Temperature in Celsius', labelFormat: '{value}C' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Climate Graph-2012'>
    <Inject services={[PolarSeries, SplineSeries, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={splineData} xName='x' yName='y'
type='Polar' name='London' drawType='Spline'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";

```

```
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    PolarSeries, Category, SplineSeries}
from '@syncfusion/ej2-react-charts';
import { splineData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { title: 'Month', valueType: 'Category' };
    const primaryyAxis: AxisModel = { minimum: -5, maximum: 35, interval: 10,
title: 'Temperature in Celsius', labelFormat: '{value}C' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Climate Graph-2012'>
        <Inject services={[PolarSeries, SplineSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={splineData} xName='x' yName='y'
type='Polar' name='London' drawType='Spline'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% enddraw %}
```

Area

To render a spline draw type in polar axis, use series [drawType](#) as **Area** and inject **AreaSeries** into services.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
PolarSeries, AreaSeries } from '@syncfusion/ej2-react-charts';
import { areaData } from 'datasource.ts';
function App() {
    const primaryxAxis = { title: 'Years', minimum: 1900, maximum: 2000,
interval: 10 };
    const primaryyAxis = { minimum: 2, maximum: 5, interval: 0.5, title:
'Sales Amount in Millions' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Average Sales Comparison'>
        <Inject services={[PolarSeries, AreaSeries]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={areaData} xName='x' yName='y'
type='Polar' name='London' drawType='Area' fill='#69D2E7'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
PolarSeries, AreaSeries }
from '@syncfusion/ej2-react-charts';
import { areaData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { title: 'Years', minimum: 1900, maximum:
2000, interval: 10 };
  const primaryyAxis: AxisModel = { minimum: 2, maximum: 5, interval: 0.5,
title: 'Sales Amount in Millions' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Average Sales Comparison'>
    <Inject services={[PolarSeries, AreaSeries]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={areaData} xName='x' yName='y'
type='Polar' name='London' drawType='Area' fill='#69D2E7'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Stacked Area

To render a stacked area draw type, use series [drawType](#) as `StackingArea` and inject `StackingAreaSeries` into services.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
PolarSeries, StackingAreaSeries, Category } from '@syncfusion/ej2-react-
charts';
import { StackedAreaData } from 'datasource.ts';
function App() {
  const primaryxAxis = { title: 'Years', valueType: 'Category',
majorGridLines: { width: 0 } };
  const primaryyAxis = { minimum: 0, maximum: 4, interval: 1, title:
'Spend Billions', majorTickLines: { width: 0 }, labelFormat: '{value}B' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Trend in Sales of Ethical Produce'>
    <Inject services={[PolarSeries, StackingAreaSeries, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={StackedAreaData} xName='x' yName='y'
type='Polar' name='Organic' drawType='StackingArea'></SeriesDirective>
      <SeriesDirective dataSource={StackedAreaData} xName='x' yName='y1'
type='Polar' name='Fair -Trade' drawType='StackingArea'></SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
```

```

        <SeriesDirective dataSource={StackedAreaData} xName='x' yName='y2'
type='Polar' name='Veg-Alternatives'
drawType='StackingArea'></SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    PolarSeries, StackingAreaSeries, Category}
from '@syncfusion/ej2-react-charts';
import { StackedAreaData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { title: 'Years', valueType: 'Category',
majorGridLines: { width: 0 } };
    const primaryyAxis: AxisModel = { minimum: 0, maximum: 4, interval: 1,
title: 'Spend Billions', majorTickLines: { width: 0 }, labelFormat:
'{$value}B' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Trend in Sales of Ethical Produce'>
        <Inject services={[PolarSeries, StackingAreaSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={StackedAreaData} xName='x' yName='y'
type='Polar' name='Organic' drawType='StackingArea'></SeriesDirective>
            <SeriesDirective dataSource={StackedAreaData} xName='x' yName='y1'
type='Polar' name='Fair -Trade' drawType='StackingArea'></SeriesDirective>
            <SeriesDirective dataSource={StackedAreaData} xName='x' yName='y2'
type='Polar' name='Veg-Alternatives'
drawType='StackingArea'></SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Column

To render a spline draw type, use series [drawType](#) as `Column` and inject `ColumnSeries` into services.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
PolarSeries, Category, ColumnSeries } from '@syncfusion/ej2-react-charts';

```

```
import { columnData } from 'datasource.ts';
function App() {
    const primaryxAxis = { title: 'Country', valueType: 'Category' };
    const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Olympic Gold medals'>
        <Inject services={[PolarSeries, ColumnSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={columnData} xName='country'
yName='gold' type='Polar' name='Gold' drawType='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    PolarSeries, Category, ColumnSeries}
from '@syncfusion/ej2-react-charts';
import { columnData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { title: 'Country', valueType: 'Category'
};
    const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Olympic Gold medals'>
        <Inject services={[PolarSeries, ColumnSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={columnData} xName='country'
yName='gold' type='Polar' name='Gold' drawType='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Stacked Column

To render a stacked column draw type, use series [drawType](#) as `StackingColumn` and inject `StackingColumnSeries` into services.

INDEX.JSX

```
{% raw %}
```

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
PolarSeries, StackingColumnSeries, Category } from '@syncfusion/ej2-react-
charts';
import { stackedColumnData } from 'datasource.ts';
function App() {
    const primaryxAxis = { title: 'Years', valueType: 'Category' };
    const primaryyAxis = { minimum: 0, maximum: 700, interval: 100, title:
'Spend Billions', labelFormat: '{value}B' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Mobile Game marker by Country'>
        <Inject services={[PolarSeries, StackingColumnSeries, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={stackedColumnData} xName='x' yName='y'
type='Polar' name='England' drawType='StackingColumn'></SeriesDirective>
            <SeriesDirective dataSource={stackedColumnData} xName='x' yName='y1'
type='Polar' name='Germany' drawType='StackingColumn'></SeriesDirective>
            <SeriesDirective dataSource={stackedColumnData} xName='x' yName='y2'
type='Polar' name='France' drawType='StackingColumn'></SeriesDirective>
            <SeriesDirective dataSource={stackedColumnData} xName='x' yName='y3'
type='Polar' name='Italy' drawType='StackingColumn'></SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    PolarSeries, StackingColumnSeries, Category}
from '@syncfusion/ej2-react-charts';
import { stackedColumnData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { title: 'Years', valueType: 'Category' };
    const primaryyAxis: AxisModel = { minimum: 0, maximum: 700, interval: 100,
title: 'Spend Billions', labelFormat: '{value}B' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Mobile Game marker by Country'>
        <Inject services={[PolarSeries, StackingColumnSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={stackedColumnData} xName='x' yName='y'
type='Polar' name='England' drawType='StackingColumn'></SeriesDirective>
            <SeriesDirective dataSource={stackedColumnData} xName='x' yName='y1'
type='Polar' name='Germany' drawType='StackingColumn'></SeriesDirective>
            <SeriesDirective dataSource={stackedColumnData} xName='x' yName='y2'
type='Polar' name='France' drawType='StackingColumn'></SeriesDirective>

```

```

        <SeriesDirective dataSource={stackedColumnData} xName='x' yName='y3'
type='Polar' name='Italy' drawType='StackingColumn'></SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% enddraw %}

```

Range Column

To render a range column draw type, use series [drawType](#) as `RangeColumn`.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
PolarSeries, RangeColumnSeries, Category } from '@syncfusion/ej2-react-
charts';
import { rangeColumnData } from 'datasource.ts';
function App() {
    const primaryxAxis = { title: 'Month', valueType: 'Category' };
    const primaryyAxis = { title: 'Temperature' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Maximum and Minimum Temperature'>
        <Inject services={[PolarSeries, RangeColumnSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective type='Polar' dataSource={rangeColumnData} xName='x'
high='high' low='low' drawType='RangeColumn'>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    PolarSeries, RangeColumnSeries, Category}
from '@syncfusion/ej2-react-charts';
import { rangeColumnData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { title: 'Month', valueType: 'Category' };
    const primaryyAxis: AxisModel = { title: 'Temperature' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Maximum and Minimum Temperature'>
        <Inject services={[PolarSeries, RangeColumnSeries, Category]} />
        <SeriesCollectionDirective>

```

```

        <SeriesDirective type='Polar' dataSource={rangeColumnData} xName='x'
high='high' low='low' drawType='RangeColumn'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Scatter

To render a scatter draw type, use series [DrawType](#) as **Scatter** and inject **ScatterSeries** module into the service.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
PolarSeries, Category, ScatterSeries } from '@syncfusion/ej2-react-charts';
import { scatterData } from 'datasource.ts';
function App() {
    const primaryxAxis = { title: 'Month', valueType: 'Category' };
    const primaryyAxis = {
        minimum: -5, maximum: 35, interval: 10, title: 'Temperature in
Celsius',
        labelFormat: '{value}C'
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Climate Graph-2012'>
        <Inject services={[PolarSeries, ScatterSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={scatterData} xName='x' yName='y'
type='Polar' name='London' drawType='Scatter'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    PolarSeries, Category, ScatterSeries}
from '@syncfusion/ej2-react-charts';
import { scatterData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { title: 'Month', valueType: 'Category' };

```



```

const primaryyAxis: AxisModel = {
  minimum: -5, maximum: 35, interval: 10, title: 'Temperature in Celsius',
  labelFormat: '{value}C'
};
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  title='Climate Graph-2012'>
  <Inject services={[PolarSeries, ScatterSeries, Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={scatterData} xName='x' yName='y'
type='Polar' name='London' drawType='Scatter'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Series customization

Start Angle

You can customize the start angle of the polar series using [startAngle](#) property. By default, `startAngle` is 0 degree.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
PolarSeries, Category, SplineSeries } from '@syncfusion/ej2-react-charts';
import { splineData } from 'datasource.ts';
function App() {
  const primaryxAxis = { title: 'Month', valueType: 'Category',
startAngle: 90 };
  const primaryyAxis = { minimum: -5, maximum: 35, interval: 10, title:
'Temperature in Celsius', labelFormat: '{value}C' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Climate Graph-2012'>
    <Inject services={[PolarSeries, SplineSeries, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={splineData} xName='x' yName='y'
type='Polar' name='London' drawType='Spline'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    PolarSeries, Category, SplineSeries}
from '@syncfusion/ej2-react-charts';
import { splineData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { title: 'Month', valueType: 'Category',
startAngle: 90 };
    const primaryyAxis: AxisModel = { minimum: -5, maximum: 35, interval: 10,
title: 'Temperature in Celsius', labelFormat: '{value}C' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Climate Graph-2012'>
        <Inject services={[PolarSeries, SplineSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={splineData} xName='x' yName='y'
type='Polar' name='London' drawType='Spline'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

Radius

You can customize the radius of the polar series using [coefficient](#) property. By default, `coefficient` is 100.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
PolarSeries, LineSeries } from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
    const primaryxAxis = { title: 'Month', coefficient: 50 };
    const primaryyAxis = { minimum: 20, maximum: 40, interval: 5, title:
'Efficiency', labelFormat: '{value}%' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Efficiency of oil-fired power
production'>
        <Inject services={[PolarSeries, LineSeries]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' type='Polar'
name='Department' drawType='Line'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
{% endraw %}
```

```

}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
PolarSeries, LineSeries }
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { title: 'Month', coefficient: 50 };
  const primaryyAxis: AxisModel = { minimum: 20, maximum: 40, interval: 5,
title: 'Efficiency', labelFormat: '{value}%' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Efficiency of oil-fired power production'>
    <Inject services={[PolarSeries, LineSeries]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' type='Polar'
name='Department' drawType='Line'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  </>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

See Also

- [Data label](#)
- [Tooltip](#)

Radar Charts in React Chart component

Radar

To render a Radar series, use series [type](#) as **Radar** and inject **RadarSeries** module into services

Draw Type

Similar to Polar drawType, Radar draw type property is used to change the series plotting type to line, column, area, range column, spline, scatter, stacking area and stacking column. The default value of drawType is **Line**.

INDEX.JSX

```

{% raw %}
import * as React from "react";

```

```

import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
RadarSeries, LineSeries } from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
    const primaryxAxis = { title: 'Month' };
    const primaryyAxis = { minimum: 20, maximum: 40, interval: 5, title:
'Efficiency', labelFormat: '{value}%' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Efficiency of oil-fired power
production'>
        <Inject services={[RadarSeries, LineSeries]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' type='Radar'
name='Department' drawType='Line'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% enddraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    RadarSeries, LineSeries}
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { title: 'Month' };
    const primaryyAxis: AxisModel = { minimum: 20, maximum: 40, interval: 5,
title: 'Efficiency', labelFormat: '{value}%' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Efficiency of oil-fired power production'>
        <Inject services={[RadarSeries, LineSeries]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' type='Radar'
name='Department' drawType='Line'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

*Series customization**Start Angle*

You can customize the start angle of the polar series using [startAngle](#) property. By default, `startAngle` is 0 degree.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
RadarSeries, LineSeries, ColumnSeries } from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
    const primaryxAxis = { title: 'Month', startAngle: 90 };
    const primaryyAxis = { minimum: 20, maximum: 40, interval: 5, title:
'Efficiency', labelFormat: '{value}%' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Efficiency of oil-fired power
production'>
        <Inject services={[RadarSeries, LineSeries, ColumnSeries]}>
            <SeriesCollectionDirective>
                <SeriesDirective dataSource={data} xName='x' yName='y' type='Radar'
drawType='Line'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    RadarSeries, LineSeries, ColumnSeries}
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { title: 'Month', startAngle: 90 };
    const primaryyAxis: AxisModel = { minimum: 20, maximum: 40, interval: 5,
title: 'Efficiency', labelFormat: '{value}%' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Efficiency of oil-fired power production'>
        <Inject services={[RadarSeries, LineSeries, ColumnSeries]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' type='Radar'
drawType='Line'>
        </SeriesDirective>
    </ChartComponent>

```

```

    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Radius

You can customize the radius of the polar series using [coefficient](#) property. By default, `coefficient` is 100.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
RadarSeries, LineSeries } from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis = { title: 'Month', coefficient: 90 };
  const primaryyAxis = { minimum: 20, maximum: 40, interval: 5, title:
'Efficiency', labelFormat: '{value}%' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Efficiency of oil-fired power
production'>
    <Inject services={[RadarSeries, LineSeries]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' type='Radar'
name='Department' drawType='Line'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
RadarSeries, LineSeries }
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { title: 'Month', coefficient: 90 };
  const primaryyAxis: AxisModel = { minimum: 20, maximum: 40, interval: 5,
title: 'Efficiency', labelFormat: '{value}%' };
  return <ChartComponent id='charts'
primaryXAxis={primaryxAxis}

```

```

    primaryYAxis={primaryyAxis}
    title='Efficiency of oil-fired power production'>
    <Inject services={[RadarSeries, LineSeries]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' type='Radar'
name='Department' drawType='Line'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

See Also

- [Data label](#)
- [Tooltip](#)

Hilo Charts in React Chart component

Hilo

To render a Hilo series, use series [type](#) as **Hilo** and inject **HiloSeries** module into the **services**.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
HiloSeries, Category, Tooltip, Zoom, Crosshair } from '@syncfusion/ej2-
react-charts';
function App() {
  const chartData = [
    { x: 'Jan', low: 87, high: 200 }, { x: 'Feb', low: 45, high: 135 },
    { x: 'Mar', low: 19, high: 85 }, { x: 'Apr', low: 31, high: 108 },
    { x: 'May', low: 27, high: 80 }, { x: 'June', low: 84, high: 130 },
    { x: 'July', low: 77, high: 150 }, { x: 'Aug', low: 54, high: 125 },
    { x: 'Sep', low: 60, high: 155 }, { x: 'Oct', low: 60, high: 180 },
    { x: 'Nov', low: 88, high: 180 }, { x: 'Dec', low: 84, high: 230 }
  ];
  const primaryxAxis = { valueType: 'Category', title: 'Months' };
  const primaryyAxis = { labelFormat: '{value}mm', edgeLabelPlacement:
'Shift', title: 'Rainfall' };
  const style = { textAlign: "center" };
  const legendSettings = { visible: false };
  return <ChartComponent id='charts' style={style}
primaryXAxis={primaryxAxis} primaryYAxis={primaryyAxis}
legendSettings={legendSettings} title='Maximum and Minimum Rainfall'>
    <Inject services={[HiloSeries, Tooltip, Category, Crosshair, Zoom]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='x' yName='low'
name='India' type='Hilo' low='low' high='high'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>

```

```

    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% enddraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
  SeriesDirective, Inject, LegendSettingsModel,
  HiloSeries, Category, Tooltip, ILoadedEventArgs, Zoom,
  Crosshair, ChartTheme }
from '@syncfusion/ej2-react-charts';
function App() {
  const chartData: any[] = [
    { x: 'Jan', low: 87, high: 200 }, { x: 'Feb', low: 45, high: 135 },
    { x: 'Mar', low: 19, high: 85 }, { x: 'Apr', low: 31, high: 108 },
    { x: 'May', low: 27, high: 80 }, { x: 'June', low: 84, high: 130 },
    { x: 'July', low: 77, high: 150 }, { x: 'Aug', low: 54, high: 125 },
    { x: 'Sep', low: 60, high: 155 }, { x: 'Oct', low: 60, high: 180 },
    { x: 'Nov', low: 88, high: 180 }, { x: 'Dec', low: 84, high: 230 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category', title: 'Months' };
  const primaryyAxis: AxisModel = { labelFormat: '{value}mm',
    edgeLabelPlacement: 'Shift', title: 'Rainfall' };
  const style: any = { textAlign: "center" };
  const legendSettings: LegendSettingsModel = { visible: false };
  return <ChartComponent id='charts' style={style}
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    legendSettings={legendSettings}
    title='Maximum and Minimum Rainfall'>
    <Inject services={[HiloSeries, Tooltip, Category, Crosshair, Zoom]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='x' yName='low'
name='India' type='Hilo' low='low'
high='high'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% enddraw %}

```

Series customization

The following properties can be used to customize the **hilo** series.

- [fill](#) – Specifies the color of the series.

- [opacity](#) – Specifies the opacity of [fill](#).

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
HiloSeries, Category, Tooltip, Zoom, Crosshair } from '@syncfusion/ej2-
react-charts';
function App() {
  const chartData = [
    { x: 'Jan', low: 87, high: 200 }, { x: 'Feb', low: 45, high: 135 },
    { x: 'Mar', low: 19, high: 85 }, { x: 'Apr', low: 31, high: 108 },
    { x: 'May', low: 27, high: 80 }, { x: 'June', low: 84, high: 130 },
    { x: 'July', low: 77, high: 150 }, { x: 'Aug', low: 54, high: 125 },
    { x: 'Sep', low: 60, high: 155 }, { x: 'Oct', low: 60, high: 180 },
    { x: 'Nov', low: 88, high: 180 }, { x: 'Dec', low: 84, high: 230 }
  ];
  const primaryxAxis = { valueType: 'Category', title: 'Months' };
  const primaryyAxis = { labelFormat: '{value}mm', edgeLabelPlacement:
'Shift', title: 'Rainfall' };
  const style = { textAlign: "center" };
  const legendSettings = { visible: false };
  return <ChartComponent id='charts' style={style}
primaryXAxis={primaryxAxis} primaryYAxis={primaryyAxis}
legendSettings={legendSettings} title='Maximum and Minimum Rainfall'>
    <Inject services={[HiloSeries, Tooltip, Category, Crosshair, Zoom]}>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='x' yName='low'
name='India' type='Hilo' low='low' high='high' fill='blue'>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel,
HiloSeries, Category, Tooltip, ILoadedEventArgs, Zoom,
Crosshair, ChartTheme }
from '@syncfusion/ej2-react-charts';
function App() {
  const chartData: any[] = [
    { x: 'Jan', low: 87, high: 200 }, { x: 'Feb', low: 45, high: 135 },
    { x: 'Mar', low: 19, high: 85 }, { x: 'Apr', low: 31, high: 108 },
    { x: 'May', low: 27, high: 80 }, { x: 'June', low: 84, high: 130 },
    { x: 'July', low: 77, high: 150 }, { x: 'Aug', low: 54, high: 125 },

```

```

    { x: 'Sep', low: 60, high: 155 }, { x: 'Oct', low: 60, high: 180 },
    { x: 'Nov', low: 88, high: 180 }, { x: 'Dec', low: 84, high: 230 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category', title: 'Months'
};
  const primaryyAxis: AxisModel = { labelFormat: '{value}mm',
edgeLabelPlacement: 'Shift', title: 'Rainfall' };
  const style: any = { textAlign: "center" };
  const legendSettings: LegendSettingsModel = { visible: false };
  return <ChartComponent id='charts' style={style}
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    legendSettings={legendSettings}
    title='Maximum and Minimum Rainfall'>
    <Inject services={[HiloSeries, Tooltip, Category, Crosshair, Zoom]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='x' yName='low'
name='India' type='Hilo' low='low'
        high='high' fill='blue'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% enddraw %}

```

See Also

- [Data label](#)
- [Tooltip](#)

High Low Open Close in React Chart component

High Low Open Close

To render a HiloOpenClose series, use series [type](#) as `HiloOpenClose` and inject `HiloOpenCloseSeries` module into the `services`.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Category, Tooltip, Zoom, Crosshair, HiloOpenCloseSeries } from
'@syncfusion/ej2-react-charts';
import { Browser } from '@syncfusion/ej2-base';
function App() {
  const chartData = [
    { x: 'Jan', open: 120, high: 160, low: 100, close: 140 },
    { x: 'Feb', open: 150, high: 190, low: 130, close: 170 },
    { x: 'Mar', open: 130, high: 170, low: 110, close: 150 },
    { x: 'Apr', open: 160, high: 180, low: 120, close: 140 },
    { x: 'May', open: 150, high: 170, low: 110, close: 130 }
  ];
  const primaryxAxis = { title: 'Date', valueType: 'Category' };

```

```

    const primaryyAxis = { title: 'Price in Dollar', minimum: 100, maximum:
200, interval: 20 };
    const style = { textAlign: "center" };
    const legendSettings = { visible: false };
    const border = { border: { width: 0 } };
    return <ChartComponent id='charts' style={style}
primaryXAxis={primaryxAxis} primaryYAxis={primaryyAxis}
legendSettings={legendSettings} chartArea={border} width={Browser.isDevice ?
'100%' : '80%'} title='Financial Analysis'>
    <Inject services={[HiloOpenCloseSeries, Tooltip, Category, Crosshair,
Zoom]} />
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={chartData} xName='x' yName='low'
name='SHIRPUR-G' type='HiloOpenClose' low='low' high='high' open='open'
close='close'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
HiloOpenCloseSeries, Selection }
from '@syncfusion/ej2-react-charts';
import { Browser } from '@syncfusion/ej2-base';
function App() {
    const chartData: any[] = [
        { x: 'Jan', open: 120, high: 160, low: 100, close: 140 },
        { x: 'Feb', open: 150, high: 190, low: 130, close: 170 },
        { x: 'Mar', open: 130, high: 170, low: 110, close: 150 },
        { x: 'Apr', open: 160, high: 180, low: 120, close: 140 },
        { x: 'May', open: 150, high: 170, low: 110, close: 130 }
    ];
    const primaryxAxis: AxisModel = { title: 'Date', valueType: 'Category' };
    const primaryyAxis: AxisModel = { title: 'Price in Dollar', minimum: 100,
maximum: 200, interval: 20 };
    const style: any = { textAlign: "center" };
    const legendSettings: LegendSettingsModel = { visible: false };
    const border = { border: { width: 0 } };
    return <ChartComponent id='charts' style={style}
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        legendSettings={legendSettings}
        chartArea={border}
        width={Browser.isDevice ? '100%' : '80%'}
        title='Financial Analysis'>
        <Inject services={[HiloOpenCloseSeries, Tooltip, Category, Crosshair,
Zoom]} />
    </ChartComponent>

```

```

    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='x' yName='low'
name='SHIRPUR-G' type='HiloOpenClose' low='low'
      high='high' open='open' close='close'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Series customization

In HiloOpenClose series, [bullFillColor](#) is used to fill the segment when the open value is greater than the close value and [bearFillColor](#) is used to fill the segment when the open value is less than the close value.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Category, Tooltip, Zoom, Crosshair, HiloOpenCloseSeries } from
'@syncfusion/ej2-react-charts';
function App() {
  const chartData = [
    { x: 'Jan', open: 120, high: 160, low: 100, close: 140 },
    { x: 'Feb', open: 150, high: 190, low: 130, close: 170 },
    { x: 'Mar', open: 130, high: 170, low: 110, close: 150 },
    { x: 'Apr', open: 160, high: 180, low: 120, close: 140 },
    { x: 'May', open: 150, high: 170, low: 110, close: 130 }
  ];
  const primaryxAxis = { title: 'Date', valueType: 'Category' };
  const primaryyAxis = { title: 'Price in Dollar', minimum: 100, maximum:
200, interval: 20 };
  const style = { textAlign: "center" };
  const legendSettings = { visible: false };
  return <ChartComponent id='charts' style={style}
primaryXAxis={primaryxAxis} primaryYAxis={primaryyAxis}
legendSettings={legendSettings} title='Financial Analysis'>
    <Inject services={[HiloOpenCloseSeries, Tooltip, Category, Crosshair,
Zoom]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='x' yName='low'
name='SHIRPUR-G' type='HiloOpenClose' low='low' high='high' open='open'
close='close' bearFillColor='#e56590' bullFillColor='#f8b883'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";

```

```
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
  SeriesDirective, Inject, LegendSettingsModel,
  Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
  HiloOpenCloseSeries, Selection }
from '@syncfusion/ej2-react-charts';
function App() {
  const chartData: any[] = [
    { x: 'Jan', open: 120, high: 160, low: 100, close: 140 },
    { x: 'Feb', open: 150, high: 190, low: 130, close: 170 },
    { x: 'Mar', open: 130, high: 170, low: 110, close: 150 },
    { x: 'Apr', open: 160, high: 180, low: 120, close: 140 },
    { x: 'May', open: 150, high: 170, low: 110, close: 130 }
  ];
  const primaryXAxis: AxisModel = { title: 'Date', valueType: 'Category' };
  const primaryYAxis: AxisModel = { title: 'Price in Dollar', minimum: 100,
maximum: 200, interval: 20 };
  const style: any = { textAlign: "center" };
  const legendSettings: LegendSettingsModel = { visible: false };
  return <ChartComponent id='charts' style={style}
    primaryXAxis={primaryXAxis}
    primaryYAxis={primaryYAxis}
    legendSettings={legendSettings}
    title='Financial Analysis'>
    <Inject services={[HiloOpenCloseSeries, Tooltip, Category, Crosshair,
Zoom]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='x' yName='low'
name='SHIRPUR-G' type='HiloOpenClose' low='low'
high='high' open='open' close='close' bearFillColor='#e56590'
bullFillColor='#f8b883'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

See Also

- [Data label](#)
- [Tooltip](#)

Candle in React Chart component

Candle

To render a Candle series, use series [type](#) as `Candle` and inject `CandleSeries` module into the `services`.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Category, Tooltip, Zoom, Crosshair, CandleSeries } from '@syncfusion/ej2-
react-charts';
function App() {
```

```

const chartData = [
  { x: 'Jan', open: 120, high: 160, low: 100, close: 140 },
  { x: 'Feb', open: 150, high: 190, low: 130, close: 170 },
  { x: 'Mar', open: 130, high: 170, low: 110, close: 150 },
  { x: 'Apr', open: 160, high: 180, low: 120, close: 140 },
  { x: 'May', open: 150, high: 170, low: 110, close: 130 }
];

const primaryxAxis = { title: 'Date', valueType: 'Category',
majorGridLines: { width: 0 } };
const primaryyAxis = { title: 'Price in Dollar', minimum: 100, maximum:
200, interval: 20 };
const style = { textAlign: "center" };
const legendSettings = { visible: false };
return <ChartComponent id='charts' style={style}
primaryXAxis={primaryxAxis} primaryYAxis={primaryyAxis}
legendSettings={legendSettings} title='Shirpur Gold Refinery Share Price'>
  <Inject services={[CandleSeries, Tooltip, Category, Crosshair,
Zoom]}/>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={chartData} xName='x' yName='low'
name='SHIRPUR-G' type='Candle' low='low' high='high' open='open'
close='close'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
CandleSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
  const chartData: any[] = [
    { x: 'Jan', open: 120, high: 160, low: 100, close: 140 },
    { x: 'Feb', open: 150, high: 190, low: 130, close: 170 },
    { x: 'Mar', open: 130, high: 170, low: 110, close: 150 },
    { x: 'Apr', open: 160, high: 180, low: 120, close: 140 },
    { x: 'May', open: 150, high: 170, low: 110, close: 130 }
  ];
  const primaryxAxis: AxisModel = { title: 'Date', valueType: 'Category',
majorGridLines: { width: 0 } };
  const primaryyAxis: AxisModel = { title: 'Price in Dollar', minimum: 100,
maximum: 200, interval: 20 };
  const style: any = { textAlign: "center" };
  const legendSettings: LegendSettingsModel = { visible: false };
  return <ChartComponent id='charts' style={style}
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}

```

```

    legendSettings={legendSettings}
    title='Shirpur Gold Refinery Share Price'>
    <Inject services={[CandleSeries, Tooltip, Category, Crosshair, Zoom]}
  />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='x' yName='low'
name='SHIRPUR-G' type='Candle' low='low'
      high='high' open='open' close='close'>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Hollow candles

Candle charts allow to visually compare the current price with previous price by coloring them.

Candles are filled/left as hollow based on the following criteria.

<!-- markdownlint-disable MD033 -->

States	Description
Filled	candle sticks are filled when the close value is lesser than the open value
Unfilled	candle sticks are unfilled when the close value is greater than the open value

The color of the candle will be defined by comparing with previous values. Bear color will be applied when the current closing value is greater than the previous closing value. Bull color will be applied when the current closing value is less than the previous closing value.

By default, bullFillColor is set as red and bearFillColor is set as green.

Solid candles

[enableSolidCandles](#) is used to enable/disable the solid candles. By default is set to be false. The fill color of the candle will be defined by its opening and closing values.

[bearFillColor](#) will be applied when the opening value is less than the closing value.

[bullFillColor](#) will be applied when the opening value is greater than closing value.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Category, Tooltip, Zoom, Crosshair, CandleSeries } from '@syncfusion/ej2-
react-charts';
function App() {
  const chartData = [
    { x: 'Jan', open: 120, high: 160, low: 100, close: 140 },
    { x: 'Feb', open: 150, high: 190, low: 130, close: 170 },
    { x: 'Mar', open: 130, high: 170, low: 110, close: 150 },
    { x: 'Apr', open: 160, high: 180, low: 120, close: 140 },
    { x: 'May', open: 150, high: 170, low: 110, close: 130 }
  ]

```

```

    ];
    const primaryxAxis = { title: 'Date', valueType: 'Category',
majorGridLines: { width: 0 } };
    const primaryyAxis = { title: 'Price in Dollar', minimum: 100, maximum:
200, interval: 20 };
    const style = { textAlign: "center" };
    const legendSettings = { visible: false };
    return <ChartComponent id='charts' style={style}
primaryXAxis={primaryxAxis} primaryYAxis={primaryyAxis}
legendSettings={legendSettings} title='Shirpur Gold Refinery Share Price'>
    <Inject services={[CandleSeries, Tooltip, Category, Crosshair,
Zoom]} />
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={chartData} xName='x' yName='low'
name='SHIRPUR-G' type='Candle' low='low' high='high' open='open'
close='close' bearFillColor='#e56590' bullFillColor='#f8b883'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel,
    Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
CandleSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
    const chartData: any[] = [
        { x: 'Jan', open: 120, high: 160, low: 100, close: 140 },
        { x: 'Feb', open: 150, high: 190, low: 130, close: 170 },
        { x: 'Mar', open: 130, high: 170, low: 110, close: 150 },
        { x: 'Apr', open: 160, high: 180, low: 120, close: 140 },
        { x: 'May', open: 150, high: 170, low: 110, close: 130 }
    ];
    const primaryxAxis: AxisModel = { title: 'Date', valueType: 'Category',
majorGridLines: { width: 0 } };
    const primaryyAxis: AxisModel = { title: 'Price in Dollar', minimum: 100,
maximum: 200, interval: 20 };
    const style: any = { textAlign: "center" };
    const legendSettings: LegendSettingsModel = { visible: false };
    return <ChartComponent id='charts' style={style}
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        legendSettings={legendSettings}
        title='Shirpur Gold Refinery Share Price'>
        <Inject services={[CandleSeries, Tooltip, Category, Crosshair, Zoom]}
/>
        <SeriesCollectionDirective>

```



```

        <SeriesDirective dataSource={chartData} xName='x' yName='low'
name='SHIRPUR-G' type='Candle' low='low'
        high='high' open='open' close='close' bearFillColor='#e56590'
bullFillColor='#f8b883'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

[See Also](#)

- [Data label](#)
- [Tooltip](#)

Box and Whisker in React Chart component

Box and Whisker

To render a box and whisker chart, use series [type](#) as `BoxAndWhisker` and inject

`BoxAndWhiskerSeries` services. The field `y` requires n number of data or it should contains minimum of five values to plot a segment.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
BoxAndWhiskerSeries, Category } from '@syncfusion/ej2-react-charts';
function App() {
    const data = [
        { x: 'Development', y: [22, 22, 23, 25, 25, 25, 26, 27, 27, 28, 28,
29, 30, 32, 34, 32, 34, 36, 35, 38] },
        { x: 'Testing', y: [22, 33, 23, 25, 26, 28, 29, 30, 34, 33, 32, 31,
50] },
        { x: 'HR', y: [22, 24, 25, 30, 32, 34, 36, 38, 39, 41, 35, 36, 40,
56] },
        { x: 'Finance', y: [26, 27, 28, 30, 32, 34, 35, 37, 35, 37, 45] },
        { x: 'R&D', y: [26, 27, 29, 32, 34, 35, 36, 37, 38, 39, 41, 43, 58]
    },
        { x: 'Sales', y: [27, 26, 28, 29, 29, 29, 32, 35, 32, 38, 53] },
        { x: 'Inventory', y: [21, 23, 24, 25, 26, 27, 28, 30, 34, 36, 38] },
        { x: 'Graphics', y: [26, 28, 29, 30, 32, 33, 35, 36, 52] },
        { x: 'Training', y: [28, 29, 30, 31, 32, 34, 35, 36] }
    ];
    const primaryxAxis = { valueType: 'Category', majorGridLines: { width: 0
    }, };
    const primaryyAxis = { minimum: 10, maximum: 60, interval: 10,
majorGridLines: { width: 0 }, majorTickLines: { width: 0 } };
    const marker = { visible: true };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Employee Age Group in Various
Department'>
        <Inject services={[Category, BoxAndWhiskerSeries]}/>
        <SeriesCollectionDirective>

```

```

        <SeriesDirective dataSource={data} xName='x' yName='y'
type='BoxAndWhisker' name='Department' marker={marker}>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
BoxAndWhiskerSeries, Category }
from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { x: 'Development', y: [22, 22, 23, 25, 25, 25, 26, 27, 27, 28, 28, 29,
30, 32, 34, 32, 34, 36, 35, 38] },
        { x: 'Testing', y: [22, 33, 23, 25, 26, 28, 29, 30, 34, 33, 32, 31, 50]
},
        { x: 'HR', y: [22, 24, 25, 30, 32, 34, 36, 38, 39, 41, 35, 36, 40, 56]
},
        { x: 'Finance', y: [26, 27, 28, 30, 32, 34, 35, 37, 35, 37, 45] },
        { x: 'R&D', y: [26, 27, 29, 32, 34, 35, 36, 37, 38, 39, 41, 43, 58] },
        { x: 'Sales', y: [27, 26, 28, 29, 29, 29, 32, 35, 32, 38, 53] },
        { x: 'Inventory', y: [21, 23, 24, 25, 26, 27, 28, 30, 34, 36, 38] },
        { x: 'Graphics', y: [26, 28, 29, 30, 32, 33, 35, 36, 52] },
        { x: 'Training', y: [28, 29, 30, 31, 32, 34, 35, 36] }
    ];
    const primaryxAxis: AxisModel = { valueType: 'Category', majorGridLines: {
width: 0 }, };
    const primaryyAxis: AxisModel = { minimum: 10, maximum: 60, interval: 10,
majorGridLines: { width: 0 }, majorTickLines: { width: 0 } };
    const marker = { visible: true };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Employee Age Group in Various Department'>
        <Inject services={[Category, BoxAndWhiskerSeries]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y'
type='BoxAndWhisker' name='Department'
                marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Box plot

You can change the rendering mode of the Box and Whisker series using the `boxPlotMode` property.

The default `boxPlotMode` is `exclusive`. The other `boxPlotMode` available are `inclusive` and `normal`.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
BoxAndWhiskerSeries, Category } from '@syncfusion/ej2-react-charts';
import { boxData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Category' };
    const primaryyAxis = { minimum: 10, maximum: 60, interval: 10 };
    const marker = { visible: true };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Employee Age Group in Various
Department'>
        <Inject services={[Category, BoxAndWhiskerSeries]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={boxData} xName='x' yName='y'
type='BoxAndWhisker' name='Department' marker={marker}>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
BoxAndWhiskerSeries, Category }
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Category' };
    const primaryyAxis: AxisModel = { minimum: 10, maximum: 60, interval: 10
};
    const marker = { visible: true };
    const boxData: any[] = [
        { x: 'Development', y: [22, 22, 23, 25, 25, 25, 26, 27, 27, 28, 28, 29,
30, 32, 34, 32, 34, 36, 35, 38] },
        { x: 'Testing', y: [22, 33, 23, 25, 26, 28, 29, 30, 34, 33, 32, 31, 50]
},
        { x: 'HR', y: [22, 24, 25, 30, 32, 34, 36, 38, 39, 41, 35, 36, 40, 56]
},
        { x: 'Finance', y: [26, 27, 28, 30, 32, 34, 35, 37, 35, 37, 45] },
        { x: 'R&D', y: [26, 27, 29, 32, 34, 35, 36, 37, 38, 39, 41, 43, 58] },
        { x: 'Sales', y: [27, 26, 28, 29, 29, 29, 32, 35, 32, 38, 53] },
        { x: 'Inventory', y: [21, 23, 24, 25, 26, 27, 28, 30, 34, 36, 38] },
        { x: 'Graphics', y: [26, 28, 29, 30, 32, 33, 35, 36, 52] },
        { x: 'Training', y: [28, 29, 30, 31, 32, 34, 35, 36] }
    ];
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Employee Age Group in Various
Department'>
        <Inject services={[Category, BoxAndWhiskerSeries]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={boxData} xName='x' yName='y'
type='BoxAndWhisker' name='Department' marker={marker}>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

```

];
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Employee Age Group in Various Department'>
        <Inject services={[Category, BoxAndWhiskerSeries]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={boxData} xName='x' yName='y'
type='BoxAndWhisker' name='Department'
                marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Show mean

In Box and Whisker series `showMean` property is used to show the box and whisker average value. The default value of `showMedian` is false.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
BoxAndWhiskerSeries, Category } from '@syncfusion/ej2-react-charts';
import { boxData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Category', majorGridLines: { width: 0
}, };
    const primaryyAxis = { minimum: 10, maximum: 60, interval: 10,
majorGridLines: { width: 0 }, majorTickLines: { width: 0 } };
    const marker = { visible: true };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Employee Age Group in Various
Department'>
        <Inject services={[Category, BoxAndWhiskerSeries]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={boxData} xName='x' yName='y'
type='BoxAndWhisker' name='Department' boxPlotMode='Normal' marker={marker}
showMean='false'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";

```

```

import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
      BoxAndWhiskerSeries, Category}
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'Category', majorGridLines: {
width: 0 }, };
  const primaryyAxis: AxisModel = { minimum: 10, maximum: 60, interval: 10,
majorGridLines: { width: 0 }, majorTickLines: { width: 0 } };
  const marker = { visible: true };
  const boxData: any[] = [
    { x: 'Development', y: [22, 22, 23, 25, 25, 25, 26, 27, 27, 28, 28, 29,
30, 32, 34, 32, 34, 36, 35, 38] },
    { x: 'Testing', y: [22, 33, 23, 25, 26, 28, 29, 30, 34, 33, 32, 31, 50]
},
    { x: 'HR', y: [22, 24, 25, 30, 32, 34, 36, 38, 39, 41, 35, 36, 40, 56]
},
    { x: 'Finance', y: [26, 27, 28, 30, 32, 34, 35, 37, 35, 37, 45] },
    { x: 'R&D', y: [26, 27, 29, 32, 34, 35, 36, 37, 38, 39, 41, 43, 58] },
    { x: 'Sales', y: [27, 26, 28, 29, 29, 29, 32, 35, 32, 38, 53] },
    { x: 'Inventory', y: [21, 23, 24, 25, 26, 27, 28, 30, 34, 36, 38] },
    { x: 'Graphics', y: [26, 28, 29, 30, 32, 33, 35, 36, 52] },
    { x: 'Training', y: [28, 29, 30, 31, 32, 34, 35, 36] }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Employee Age Group in Various Department'>
    <Inject services={[Category, BoxAndWhiskerSeries]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={boxData} xName='x' yName='y'
type='BoxAndWhisker' name='Department' boxPlotMode='Normal'
        marker={marker} showMean=false>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

See Also

- [Data label](#)
- [Tooltip](#)

Waterfall in React Chart component

Waterfall chart

To render a waterfall series, use series `type` as `Waterfall` and inject `WaterfallSeries` module into the services.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";

```

```

import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, WaterfallSeries }
from '@syncfusion/ej2-react-charts';
function App() {
  const data = [
    { x: 'Income', y: 4711 }, { x: 'Sales', y: -1015 },
    { x: 'Development', y: -688 },
    { x: 'Revenue', y: 1030 }, { x: 'Balance' },
    { x: 'Expense', y: -361 }, { x: 'Tax', y: -695 },
    { x: 'Net Profit' }
  ];
  const primaryxAxis = { valueType: 'Category', majorGridLines: { width: 0
}, plotOffset: 20 };
  const primaryyAxis = { minimum: 0, maximum: 5000, interval: 1000,
majorGridLines: { width: 0 }, title: 'Expenditure' };
  const marker = { dataLabel: { visible: true, font: { color: '#ffffff' }
} };
  const connector = { color: '#5F6A6A', width: 2 };
  const tooltip = { enable: true, shared: false };
  const legendSettings = { visible: false };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip}
legendSettings={legendSettings} title='Company Revenue and Profit'>
    <Inject services={[WaterfallSeries, Category, Tooltip, Zoom,
Crosshair, Legend, DataLabel]}>
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y' name='USA'
type='Waterfall' intermediateSumIndexes={[4]} sumIndexes={[7]}
marker={marker} connector={connector} columnWidth={0.9}
negativeFillColor='#e56590'>
          </SeriesDirective>
        </SeriesCollectionDirective>
      </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, TooltipSettingsModel, LegendSettingsModel,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
WaterfallSeries, Selection }
from '@syncfusion/ej2-react-charts';
function App() {
  const data: object[] = [
    { x: 'Income', y: 4711 }, { x: 'Sales', y: -1015 },
    { x: 'Development', y: -688 },
    { x: 'Revenue', y: 1030 }, { x: 'Balance' },
    { x: 'Expense', y: -361 }, { x: 'Tax', y: -695 },
    { x: 'Net Profit' }
  ];
};

```

```

const primaryxAxis: AxisModel = { valueType: 'Category', majorGridLines: {
width: 0 }, plotOffset: 20 };
const primaryyAxis: AxisModel = { minimum: 0, maximum: 5000, interval:
1000, majorGridLines: { width: 0 }, title: 'Expenditure' };
const marker = { dataLabel: { visible: true, font: { color: '#ffffff' } }
};
const connector = { color: '#5F6A6A', width: 2 };
const tooltip: TooltipSettingsModel = { enable: true, shared: false };
const legendSettings: LegendSettingsModel = { visible: false };
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  tooltip={tooltip}
  legendSettings={legendSettings}
  title='Company Revenue and Profit'>
  <Inject services=[WaterfallSeries, Category, Tooltip, Zoom,
Crosshair, Legend, DataLabel]] />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y' name='USA'
type='Waterfall' intermediateSumIndexes={[4]}
    sumIndexes={[7]} marker={marker} connector={connector}
columnWidth={0.9}
    negativeFillColor='#e56590'>
  </SeriesDirective>
</SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Series customization

The negative changes of waterfall series is represented by using [negativeFillColor](#) and the summary changes are represented by using [summaryFillColor](#) properties.

By default, the negativeFillColor as '#E94649' and the summaryFillColor as '#4E81BC'.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, WaterfallSeries }
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis = { crossesAt: 15 };
  const primaryyAxis = { crossesAt: 5 };
  const marker = { dataLabel: { visible: true, font: { color: '#ffffff' } }
  };
  const connector = { color: '#5F6A6A', width: 2 };
  const tooltip = { enable: true, shared: false };
  const legendSettings = { visible: false };
  return <ChartComponent id='charts' primaryXAxis={{ valueType:
'Category', majorGridLines: { width: 0 }, plotOffset: 20 }} primaryYAxis={{
minimum: 0, maximum: 5000, interval: 1000, majorGridLines: { width: 0 },

```

```

title: 'Expenditure' }} tooltip={tooltip} legendSettings={legendSettings}
title='Company Revenue and Profit'>
  <Inject services={[WaterfallSeries, Category, Tooltip, Zoom,
Crosshair, Legend, DataLabel]}/>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y' name='USA'
type='Waterfall' intermediateSumIndexes={[4]} sumIndexes={[7]}
marker={marker} connector={connector} columnWidth={0.9}
summaryFillColor='#e56590' negativeFillColor='#f8b883'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, TooltipSettingsModel, LegendSettingsModel,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
WaterfallSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { crossesAt: 15 };
  const primaryyAxis: AxisModel = { crossesAt: 5 };
  const marker = { dataLabel: { visible: true, font: { color: 'ffffff' } } };
};
const connector = { color: '#5F6A6A', width: 2 };
const tooltip: TooltipSettingsModel = { enable: true, shared: false };
const legendSettings: LegendSettingsModel = { visible: false };
const data: any[] = [
  { x: 'Income', y: 4711 }, { x: 'Sales', y: -1015 },
  { x: 'Development', y: -688 },
  { x: 'Revenue', y: 1030 }, { x: 'Balance' },
  { x: 'Expense', y: -361 }, { x: 'Tax', y: -695 },
  { x: 'Net Profit' }
];
return <ChartComponent id='charts'
  primaryXAxis={{ valueType: 'Category', majorGridLines: { width: 0 },
plotOffset: 20 }}
  primaryYAxis={{ minimum: 0, maximum: 5000, interval: 1000,
majorGridLines: { width: 0 }, title: 'Expenditure' }}
  tooltip={tooltip}
  legendSettings={legendSettings}
  title='Company Revenue and Profit'>
  <Inject services={[WaterfallSeries, Category, Tooltip, Zoom,
Crosshair, Legend, DataLabel]}/>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y' name='USA'
type='Waterfall' intermediateSumIndexes={[4]}

```



```

        sumIndexes={[7]} marker={marker} connector={connector}
        columnWidth={0.9}
        summaryFillColor='#e56590' negativeFillColor='#f8b883'>
    </SeriesDirective>
</SeriesCollectionDirective>
</ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% enddraw %}

```

See Also

- [Data label](#)
- [Tooltip](#)

Histogram in React Chart component

Histogram Series

Histogram type charts can provide a visual display of large amounts of data that are difficult to understand in a tabular or spreadsheet form.

To render a histogram chart, use series [type](#) as `Histogram` and inject `HistogramSeries` module into the services.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, HistogramSeries } from
"@syncfusion/ej2-react-charts";
function App() {
    const chartData = [];
    const points = [5.250, 7.750, 0, 8.275, 9.750, 7.750, 8.275, 6.250,
5.750,
    5.250, 23.000, 26.500, 27.750, 25.025, 26.500, 26.500, 28.025,
29.250, 26.750, 27.250,
    26.250, 25.250, 34.500, 25.625, 25.500, 26.625, 36.275, 36.250,
26.875, 40.000, 43.000,
    46.500, 47.750, 45.025, 56.500, 56.500, 58.025, 59.250, 56.750,
57.250,
    46.250, 55.250, 44.500, 45.525, 55.500, 46.625, 46.275, 56.250,
46.875, 43.000,
    46.250, 55.250, 44.500, 45.425, 55.500, 56.625, 46.275, 56.250,
46.875, 43.000,
    46.250, 55.250, 44.500, 45.425, 55.500, 46.625, 56.275, 46.250,
56.875, 41.000, 63.000,
    66.500, 67.750, 65.025, 66.500, 76.500, 78.025, 79.250, 76.750,
77.250,
    66.250, 75.250, 74.500, 65.625, 75.500, 76.625, 76.275, 66.250,
66.875, 80.000, 85.250,
    87.750, 89.000, 88.275, 89.750, 97.750, 98.275, 96.250, 95.750,
95.250

```

```

];
function chartLoad() {
  points.map((value) => {
    chartData.push({
      y: value
    });
  });
}
const primaryxAxis = { majorGridLines: { width: 0 }, title: 'Score of
Final Examination', minimum: 0, maximum: 100 };
const primaryyAxis = { title: 'Number of Students', minimum: 0, maximum:
50, interval: 10, majorTickLines: { width: 0 }, lineStyle: { width: 0 } };
const legendSettings = { visible: false };
const tooltipSettings = { enable: true };
const marker = { dataLabel: { visible: true, position: 'Top', font: {
fontWeight: '600', color: '#ffffff' } } };
chartLoad();
return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltipSettings}
legendSettings={legendSettings} title='Examination Results'>
  <Inject services={[HistogramSeries, Legend, Tooltip,
Category, DataLabel]}/>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={chartData} yName='y'
name='Score' type='Histogram' marker={marker} showNormalDistribution={true}
columnWidth={0.99} binInterval={20}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
  AxisModel, ChartComponent, SeriesCollectionDirective, SeriesDirective,
  Inject, ChartTheme, LegendSettingsModel, TooltipSettingsModel,
  Legend, Category, Tooltip, ColumnSeries, ILoadedEventArgs, DataLabel,
  HistogramSeries
} from '@syncfusion/ej2-react-charts';
function App() {
  const chartData: Object[] = [];
  const points: number[] = [5.250, 7.750, 0, 8.275, 9.750, 7.750, 8.275,
6.250, 5.750,
  5.250, 23.000, 26.500, 27.750, 25.025, 26.500, 26.500, 28.025,
29.250, 26.750, 27.250,
  26.250, 25.250, 34.500, 25.625, 25.500, 26.625, 36.275, 36.250,
26.875, 40.000, 43.000,
  46.500, 47.750, 45.025, 56.500, 56.500, 58.025, 59.250, 56.750,
57.250,

```

```

46.250, 55.250, 44.500, 45.525, 55.500, 46.625, 46.275, 56.250,
46.875, 43.000,
46.250, 55.250, 44.500, 45.425, 55.500, 56.625, 46.275, 56.250,
46.875, 43.000,
46.250, 55.250, 44.500, 45.425, 55.500, 46.625, 56.275, 46.250,
56.875, 41.000, 63.000,
66.500, 67.750, 65.025, 66.500, 76.500, 78.025, 79.250, 76.750,
77.250,
66.250, 75.250, 74.500, 65.625, 75.500, 76.625, 76.275, 66.250,
66.875, 80.000, 85.250,
87.750, 89.000, 88.275, 89.750, 97.750, 98.275, 96.250, 95.750,
95.250
];
function chartLoad(): void {
    points.map((value: number) => {
        chartData.push({
            y: value
        });
    });
}
const primaryxAxis: AxisModel= { majorGridLines: { width: 0 }, title:
'Score of Final Examination', minimum: 0, maximum: 100 } ;
const primaryyAxis: AxisModel= { title: 'Number of Students', minimum:
0, maximum: 50, interval: 10, majorTickLines: { width: 0 }, lineStyle: {
width: 0 } } ;
const legendSettings: LegendSettingsModel= { visible: false };
const tooltipsettings: TooltipSettingsModel={ enable: true };
const marker={ dataLabel: { visible: true, position: 'Top', font: {
fontWeight: '600', color: '#ffffff' } } };
chartLoad();
return <ChartComponent id='charts'
    primaryXAxis={ primaryxAxis }
    primaryYAxis={ primaryyAxis }
    tooltip={ tooltipsettings }
    legendSettings={ legendSettings }
    title='Examination Results'>
    <Inject services=[HistogramSeries, Legend, Tooltip,
Category, DataLabel]] />
    <SeriesCollectionDirective>
    <SeriesDirective dataSource={chartData} yName='y'
name='Score' type='Histogram'
        marker={ marker }
        showNormalDistribution={true} columnWidth={0.99}
binInterval={20}>
        </SeriesDirective>
    </SeriesCollectionDirective>
    </ChartComponent>
</>
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

See Also

- [Data label](#)

- [Tooltip](#)

Error Bar in React Chart component

Error Bar

Error bars are graphical representations of the variability of data and used on graphs to indicate the error or uncertainty in a reported measurement. To render the error bar for the series, set [visible](#) as `true` and inject `ErrorBar` module into the `services`.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, LineSeries, ErrorBar } from '@syncfusion/ej2-react-
charts';
function App() {
    const data = [
        { x: 2006, y: 7.8 }, { x: 2007, y: 7.2 },
        { x: 2008, y: 6.8 }, { x: 2009, y: 10.7 },
        { x: 2010, y: 10.8 }, { x: 2011, y: 9.8 }
    ];
    const primaryxAxis = { minimum: 2005, maximum: 2012, interval: 1, title:
'Year' };
    const primaryyAxis = { minimum: 3, maximum: 12, interval: 1, title:
'Percentage', labelFormat: '{value}%' };
    const marker;
    const errorbar = { visible: true };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Unemployment rate (%)'>
        <Inject services={[LineSeries, Legend, Category, ErrorBar]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India' type='Line' marker={marker} errorBar={errorbar}>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
        Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries,
Selection, ErrorBar }
from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
```

```

    { x: 2006, y: 7.8 }, { x: 2007, y: 7.2 },
    { x: 2008, y: 6.8 }, { x: 2009, y: 10.7 },
    { x: 2010, y: 10.8 }, { x: 2011, y: 9.8 }
  ];
  const primaryxAxis: AxisModel = { minimum: 2005, maximum: 2012, interval:
1, title: 'Year' };
  const primaryyAxis: AxisModel = { minimum: 3, maximum: 12, interval: 1,
title: 'Percentage', labelFormat: '{value}%' };
  const marker: { visible: true };
  const errorbar = { visible: true };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Unemployment rate (%)'>
    <Inject services={[LineSeries, Legend, Category, ErrorBar]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India' type='Line'
        marker={marker} errorBar={errorbar}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Error Bar Type

To change the error bar rendering type using [type](#) option of error bar. To change the error bar line length you can use [verticalError](#) property.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, LineSeries, ErrorBar } from '@syncfusion/ej2-react-
charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis = { minimum: 2005, maximum: 2012, interval: 1, title:
'Year' };
  const primaryyAxis = { minimum: 3, maximum: 12, interval: 1, title:
'Percentage', labelFormat: '{value}%' };
  const marker;
  const errorbar = { visible: true, type: 'Percentage', verticalError: 4
};
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Unemployment rate (%)'>
    <Inject services={[LineSeries, Legend, Category, ErrorBar]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India' type='Line' marker={marker} errorBar={errorbar}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

```

    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries,
Selection, ErrorBar }
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { minimum: 2005, maximum: 2012, interval:
1, title: 'Year' };
    const primaryyAxis: AxisModel = { minimum: 3, maximum: 12, interval: 1,
title: 'Percentage', labelFormat: '{value}%' };
    const marker: { visible: true };
    const errorbar = { visible: true, type: 'Percentage', verticalError: 4 };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Unemployment rate (%)'>
        <Inject services={[LineSeries, Legend, Category, ErrorBar]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India' type='Line'
                marker={marker} errorBar={errorbar}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Customizing error bar type

To customize the error bar type, set error bar [type](#) as **Custom** and then change the horizontal/vertical positive and negative error of error bar.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, LineSeries, ErrorBar } from '@syncfusion/ej2-react-
charts';
import { data } from 'datasource.ts';

```

```
function App() {
  const primaryxAxis = { minimum: 2005, maximum: 2012, interval: 1, title:
'Year' };
  const primaryyAxis = { minimum: 3, maximum: 12, interval: 1, title:
'Percentage', labelFormat: '{value}%' };
  const marker;
  const errorbar = {
    visible: true,
    type: 'Custom',
    mode: 'Both',
    verticalPostiveError: 3,
    horizontalPositiveError: 2,
    verticalNegativeError: 3,
    horizontalNegativeError: 2
  };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Unemployment rate (%)'>
    <Inject services={[LineSeries, Legend, Category, ErrorBar]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India' type='Line' marker={marker} errorBar={errorbar}>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
  Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries,
Selection, ErrorBar }
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { minimum: 2005, maximum: 2012, interval:
1, title: 'Year' };
  const primaryyAxis: AxisModel = { minimum: 3, maximum: 12, interval: 1,
title: 'Percentage', labelFormat: '{value}%' };
  const marker: { visible: true };
  const errorbar = {
    visible: true,
    type: 'Custom',
    mode: 'Both',
    verticalPostiveError: 3,
    horizontalPositiveError: 2,
    verticalNegativeError: 3,
    horizontalNegativeError: 2
  };
};
```

```

return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  title='Unemployment rate (%)'>
  <Inject services={[LineSeries, Legend, Category, ErrorBar]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India' type='Line'
      marker={marker} errorBar={errorbar}>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Error bar mode

Error bar mode is used to define whether the error bar line has to be drawn horizontally, vertically or in both side. To change the error bar mode use [mode](#) option.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, LineSeries, ErrorBar } from '@syncfusion/ej2-react-
charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis = { minimum: 2005, maximum: 2012, interval: 1, title:
'Year' };
  const primaryyAxis = { minimum: 3, maximum: 12, interval: 1, title:
'Percentage', labelFormat: '{value}%' };
  const marker;
  const errorbar = { visible: true, mode: 'Horizontal' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Unemployment rate (%)'>
    <Inject services={[LineSeries, Legend, Category, ErrorBar]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India' type='Line' marker={marker} errorBar={errorbar}>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}

```



```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries,
Selection, ErrorBar }
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { minimum: 2005, maximum: 2012, interval:
1, title: 'Year' };
  const primaryyAxis: AxisModel = { minimum: 3, maximum: 12, interval: 1,
title: 'Percentage', labelFormat: '{value}%' };
  const marker: { visible: true };
  const errorbar = { visible: true, mode: 'Horizontal' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Unemployment rate (%)'>
    <Inject services={[LineSeries, Legend, Category, ErrorBar]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India' type='Line'
        marker={marker} errorBar={errorbar}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Error bar direction

To change the error bar direction to plus, minus or both side using [direction](#) option.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, LineSeries, ErrorBar } from '@syncfusion/ej2-react-
charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis = { minimum: 2005, maximum: 2012, interval: 1, title:
'Year' };
  const primaryyAxis = { minimum: 3, maximum: 12, interval: 1, title:
'Percentage', labelFormat: '{value}%' };
  const marker;
  const errorbar = { visible: true, mode: 'Vertical', direction: 'Minus'
};
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Unemployment rate (%)'>
    <Inject services={[LineSeries, Legend, Category, ErrorBar]} />

```

```

        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India' type='Line' marker={marker} errorBar={errorbar}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries,
Selection, ErrorBar }
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { minimum: 2005, maximum: 2012, interval:
1, title: 'Year' };
    const primaryyAxis: AxisModel = { minimum: 3, maximum: 12, interval: 1,
title: 'Percentage', labelFormat: '{value}%' };
    const marker: { visible: true };
    const errorbar = { visible: true, mode: 'Vertical', direction: 'Minus' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Unemployment rate (%)'>
        <Inject services={[LineSeries, Legend, Category, ErrorBar]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India' type='Line'
                marker={marker} errorBar={errorbar}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Customizing error bar cap

To customize the error bar cap length, width and fill color, you can use [errorBarCap](#) option.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";

```

```

import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, LineSeries, ErrorBar } from '@syncfusion/ej2-react-
charts';
import { data } from 'datasource.ts';
function App() {
    const primaryxAxis = { minimum: 2005, maximum: 2012, interval: 1, title:
'Year' };
    const primaryyAxis = { minimum: 3, maximum: 12, interval: 1, title:
'Percentage', labelFormat: '{value}%' };
    const marker;
    const errorbar = {
        visible: true, errorBarCap: {
            length: 10,
            width: 10,
            color: '#0000ff'
        }
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Unemployment rate (%)'>
        <Inject services={[LineSeries, Legend, Category, ErrorBar]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India' type='Line' marker={marker} errorBar={errorbar}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries,
Selection, ErrorBar }
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { minimum: 2005, maximum: 2012, interval:
1, title: 'Year' };
    const primaryyAxis: AxisModel = { minimum: 3, maximum: 12, interval: 1,
title: 'Percentage', labelFormat: '{value}%' };
    const marker: { visible: true };
    const errorbar = {
        visible: true, errorBarCap: {
            length: 10,
            width: 10,
            color: '#0000ff'
        }
    };
};

```

```

return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  title='Unemployment rate (%)'>
  <Inject services={[LineSeries, Legend, Category, ErrorBar]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India' type='Line'
      marker={marker} errorBar={errorbar}>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Customizing error bar color

To customize the error bar color for individual errors, use the [errorBarColorMapping](#) property. You can also customize the vertical error, horizontal error, horizontal negative and positive error and vertical negative and positive error for an individual point using [verticalError](#), [horizontalError](#), [horizontalNegativeError](#), [horizontalPositiveError](#), [verticalNegativeError](#) and [verticalPositiveError](#) properties.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, LineSeries, ErrorBar } from '@syncfusion/ej2-react-
charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis = { minimum: 2005, maximum: 2012, interval: 1, title:
'Year' };
  const primaryyAxis = { minimum: 3, maximum: 12, interval: 1, title:
'Percentage', labelFormat: '{value}%' };
  const errorbar = {
    visible: true, errorBarColorMapping: 'color', verticalError: 'error'
  };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Unemployment rate (%)'>
    <Inject services={[LineSeries, Legend, Category, ErrorBar]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
type='Line' errorBar={errorbar}>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
      Legend, Category, LineSeries, ErrorBar }
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { minimum: 2005, maximum: 2012, interval:
1, title: 'Year' };
  const primaryyAxis: AxisModel = { minimum: 3, maximum: 12, interval: 1,
title: 'Percentage', labelFormat: '{value}%' };
  const errorbar = {
    visible: true, errorBarColorMapping: 'color', verticalError: 'error'
  };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Unemployment rate (%)'>
    <Inject services={[LineSeries, Legend, Category, ErrorBar]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
type='Line'
        errorBar={errorbar}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

See Also

- [Data label](#)
- [Tooltip](#)

Vertical Chart in React Chart component

Vertical

In EJ2 chart, you can draw a chart in vertical manner by changing orientation of the axis. All series types support this feature. You can use `isTransposed` property in chart to render a chart in vertical manner.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, SplineSeries } from '@syncfusion/ej2-
react-charts';
```

```

import { splineData } from 'datasource.ts';
function App() {
  const primaryxAxis = { title: 'Month', valueType: 'Category' };
  const primaryyAxis = { minimum: -5, maximum: 35, interval: 5, title:
'Temperature in Celsius', labelFormat: '{value}C' };
  const marker;
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='CO2 - Intensity Analysis'
isTransposed={true}>
    <Inject services={[SplineSeries, Legend, Tooltip,
DataLabel, Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={splineData} xName='x'
yName='y' width={2} name='London' type='Spline' marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
SplineSeries, Selection}
from '@syncfusion/ej2-react-charts';
import { splineData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel= { title: 'Month', valueType: 'Category' }
;
  const primaryyAxis: AxisModel= { minimum: -5, maximum: 35, interval:
5, title: 'Temperature in Celsius', labelFormat: '{value}C' } ;
  const marker:{ visible: true, width: 10, height: 10 };
  return <ChartComponent id='charts'
    primaryXAxis={ primaryxAxis }
    primaryYAxis={ primaryyAxis }
    title='CO2 - Intensity Analysis' isTransposed={ true }>
    <Inject services={[SplineSeries, Legend, Tooltip,
DataLabel, Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource ={splineData} xName='x'
yName='y' width={2} name='London' type='Spline'
marker={ marker }>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

```
{% enddraw %}
```

See Also

- [Data label](#)
- [Tooltip](#)

Pareto in React Chart component

Pareto

Pareto charts are used to find the cumulative values of data in different categories. It is a combination of Column and Line series.

The initial values are represented by column chart and the cumulative values are represented by Line chart.

To render a pareto chart, use series [type](#) as

Pareto and inject `ParetoSeries` `ColumnSeries` and `LineSeries` module.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ParetoSeries, ColumnSeries, LineSeries, Legend, Category, Tooltip, DataLabel
} from '@syncfusion/ej2-react-charts';
function App() {
    const chartData = [
        { x: 'Traffic', y: 56 }, { x: 'Child Care', y: 44.8 },
        { x: 'Transport', y: 27.2 }, { x: 'Weather', y: 19.6 },
        { x: 'Emergency', y: 6.6 }
    ];
    const primaryxAxis = { title: 'Defects', valueType: 'Category' };
    const primaryyAxis = { title: 'Frequency', minimum: 0, maximum: 150,
interval: 30 };
    const tooltipsettings = { enable: true };
    const marker = { visible: true, width: 10, height: 10 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltipsettings} title='Defect vs
Frequency'>
        <Inject services={[ColumnSeries, LineSeries,
ParetoSeries, Legend, Tooltip, Category, DataLabel]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} xName='x'
yName='y' name='Defect' type='Pareto' marker={marker}>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
    }
    ;
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import {
  AxisModel, ChartComponent, SeriesCollectionDirective, SeriesDirective,
  Inject,
  ChartTheme, LegendSettingsModel, TooltipSettingsModel, ParetoSeries, ColumnSeries,
  LineSeries,
  Legend, Category, Tooltip, ILoadedEventArgs, DataLabel
} from '@syncfusion/ej2-react-charts';
function App() {
  const chartData: any[]=[
    { x: 'Traffic', y: 56 }, { x: 'Child Care', y: 44.8 },
    { x: 'Transport', y: 27.2 }, { x: 'Weather', y: 19.6 },
    { x: 'Emergency', y: 6.6 }
  ];
  const primaryxAxis: AxisModel= { title: 'Defects', valueType:
'Category' } ;
  const primaryyAxis: AxisModel= { title: 'Frequency', minimum: 0,
maximum: 150, interval: 30 } ;
  const tooltipsettings: TooltipSettingsModel={ enable: true };
  const marker={ visible: true, width: 10, height: 10};
  return <ChartComponent id='charts'
    primaryXAxis={ primaryxAxis }
    primaryYAxis={ primaryyAxis }
    tooltip={ tooltipsettings }
    title='Defect vs Frequency'>
    <Inject services={[ColumnSeries, LineSeries,
ParetoSeries, Legend, Tooltip, Category, DataLabel]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} xName='x'
yName='y' name='Defect' type='Pareto'
        marker={ marker }>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Pareto customization

The pareto line series can be customized by using the [marker](#), [width](#), [dashArray](#), and [fill](#) properties in the [paretoOptions](#). The secondary axis for the pareto series can be shown or hidden using the [showAxis](#) property.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ParetoSeries, ColumnSeries, LineSeries, Legend, Category, Tooltip, DataLabel
} from '@syncfusion/ej2-react-charts';
function App() {
  const chartData = [
    { x: 'Button Defect', y: 23 }, { x: 'Pocket Defect', y: 16 },

```



```

        { x: 'Collar Defect ', y: 10 }, { x: 'Cuff Defect', y: 7 },
        { x: 'Sleeve Defect', y: 6 }, { x: 'Other Defect', y: 2 }
    ];
    const primaryxAxis = { interval: 1, valueType: 'Category',
    majorGridLines: { width: 0 }, labelIntersectAction: 'Rotate45',
    minorGridLines: { width: 0 }, majorTickLines: { width: 0 }, minorTickLines:
    { width: 0 }, lineStyle: { width: 0 } };
    const primaryyAxis = { title: 'Frequency of Occurence', minimum: 0,
    maximum: 25, interval: 5, lineStyle: { width: 0 }, majorTickLines: { width:
    0 }, majorGridLines: { width: 1 }, minorGridLines: { width: 1 },
    minorTickLines: { width: 0 } };
    const tooltipsettings = { enable: true, shared: true, format:
    '${series.name} : <b>${point.y}</b>' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        tooltip={tooltipsettings}
        title='Defects in Shirts'
        legendSettings={{ visible: true, enableHighlight: true }}>
        <Inject services={[ColumnSeries, LineSeries, ParetoSeries, Legend,
        Tooltip, Category, DataLabel, Highlight]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} xName='x' yName='y'
            name='Defect' type='Pareto' width={2} opacity={0.75} columnWidth={0.4}
            cornerRadius={{ topLeft: Browser.isDevice ? 4 : 6, topRight:
            Browser.isDevice ? 4 : 6 }} paretoOptions={{ marker: { visible: true,
            isFilled: true, width: 7, height: 7 }, dashArray: '3,2', width: 2, fill:
            '#F7523F' }}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
    AxisModel, ChartComponent, SeriesCollectionDirective, SeriesDirective,
    Inject, ChartTheme, LegendSettingsModel, TooltipSettingsModel, ParetoSeries,
    ColumnSeries, LineSeries,
    Legend, Category, Tooltip, ILoadedEventArgs, DataLabel, Highlight
} from '@syncfusion/ej2-react-charts';
import { Browser } from '@syncfusion/ej2-base';
function App() {
    const chartData: any[] = [
        { x: 'Button Defect', y: 23 }, { x: 'Pocket Defect', y: 16 },
        { x: 'Collar Defect ', y: 10 }, { x: 'Cuff Defect', y: 7 },
        { x: 'Sleeve Defect', y: 6 }, { x: 'Other Defect', y: 2 }
    ];

```

```

const primaryxAxis: AxisModel = { interval: 1, valueType: 'Category',
majorGridLines: { width: 0 }, labelIntersectAction: 'Rotate45',
minorGridLines: { width: 0 }, majorTickLines: { width: 0 }, minorTickLines:
{ width: 0 }, lineStyle: { width: 0 } };
const primaryyAxis: AxisModel = { title: 'Frequency of Occurrence',
minimum: 0, maximum: 25, interval: 5, lineStyle: { width: 0 },
majorTickLines: { width: 0 }, majorGridLines: { width: 1 }, minorGridLines:
{ width: 1 }, minorTickLines: { width: 0 } };
const tooltipsettings: TooltipSettingsModel = { enable: true, shared:
true, format: '${series.name} : <b>${point.y}</b>' };
return <ChartComponent id='charts'
primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis}
tooltip={tooltipsettings}
title='Defects in Shirts'
legendSettings={{ visible: true, enableHighlight: true }}>
<Inject services={[ColumnSeries, LineSeries, ParetoSeries, Legend,
Tooltip, Category, DataLabel, Highlight]} />
<SeriesCollectionDirective>
<SeriesDirective dataSource={chartData} xName='x' yName='y'
name='Defect' type='Pareto' width={2} opacity={0.75} columnWidth={0.4}
cornerRadius={{ topLeft: Browser.isDevice ? 4 : 6, topRight:
Browser.isDevice ? 4 : 6 }} paretoOptions={{ marker: { visible: true,
isFilled: true, width: 7, height: 7 }, dashArray: '3,2', width: 2, fill:
'#F7523F' }}>
</SeriesDirective>
</SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

See also

- [Data label](#)
- [Tooltip](#)

Chart series in React Chart component

Multiple Series

You can add multiple series to the chart by using [series](#) property. The series are rendered in the order as it is added to the series array.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, LineSeries } from
"@syncfusion/ej2-react-charts";
function App() {
const data = [
{ country: "USA", gold: 50, silver: 70, bronze: 45 },
{ country: "China", gold: 40, silver: 60, bronze: 55 },

```

```

    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  const primaryxAxis = { valueType: 'Category', title: 'Countries' };
  const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='silver'
type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='bronze'
type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
  const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}

```

```

    primaryYAxis={primaryyAxis}
    title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, Category]} />
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='country' yName='gold'
type='Column'>
        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='country' yName='silver'
type='Column'>
        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='country' yName='bronze'
type='Column'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Combination Series

Combination of different types like Line, column etc, can be rendered in a chart.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, LineSeries, StackingColumnSeries }
from '@syncfusion/ej2-react-charts';
function App() {
    const data = [
        { x: '2005', y: 1.2, y1: 0.5, y2: 0.7, y3: -0.8, y4: 1.5 },
        { x: '2006', y: 1, y1: 0.5, y2: 1.4, y3: 0, y4: 2.3 },
        { x: '2007', y: 1, y1: 0.5, y2: 1.5, y3: -1, y4: 2 },
        { x: '2008', y: 0.25, y1: 0.35, y2: 0.35, y3: -.35, y4: 0.1 },
        { x: '2009', y: 0.1, y1: 0.9, y2: -2.7, y3: -0.3, y4: -2.7 },
        { x: '2010', y: 1, y1: 0.5, y2: 0.5, y3: -0.5, y4: 1.8 },
        { x: '2011', y: 0.1, y1: 0.25, y2: 0.25, y3: 0, y4: 2 },
        { x: '2012', y: -0.25, y1: -0.5, y2: -0.1, y3: -0.4, y4: 0.4 },
        { x: '2013', y: 0.25, y1: 0.5, y2: -0.3, y3: 0, y4: 0.9 },
        { x: '2014', y: 0.6, y1: 0.6, y2: -0.6, y3: -0.6, y4: 0.4 },
        { x: '2015', y: 0.9, y1: 0.5, y2: 0, y3: -0.3, y4: 1.3 }
    ];
    const primaryxAxis = { title: 'Years', interval: 1,
labelIntersectAction: 'Rotate45', valueType: 'Category' };
    const primaryyAxis = { title: 'Growth', minimum: -3, maximum: 3,
interval: 1 };
    const marker = { visible: true, width: 10, opacity: 0.6, height: 10 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Annual Growth GDP in France'>
        <Inject services={[StackingColumnSeries, LineSeries, Legend, Tooltip,
DataLabel, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' name='Private
Consumption' type='StackingColumn'>

```

```

        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y1'
name='Government Consumption' type='StackingColumn'>
        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y2'
name='Investment' type='StackingColumn'>
        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y3' name='Net
Foreign Trade' type='StackingColumn'>
        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y4' name='GDP'
type='Line' opacity={0.6} marker={marker}>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    Legend, Category, Tooltip, DataLabel, LineSeries,
StackingColumnSeries}
from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { x: '2005', y: 1.2, y1: 0.5, y2: 0.7, y3: -0.8, y4: 1.5 },
        { x: '2006', y: 1, y1: 0.5, y2: 1.4, y3: 0, y4: 2.3 },
        { x: '2007', y: 1, y1: 0.5, y2: 1.5, y3: -1, y4: 2 },
        { x: '2008', y: 0.25, y1: 0.35, y2: 0.35, y3: -.35, y4: 0.1 },
        { x: '2009', y: 0.1, y1: 0.9, y2: -2.7, y3: -0.3, y4: -2.7 },
        { x: '2010', y: 1, y1: 0.5, y2: 0.5, y3: -0.5, y4: 1.8 },
        { x: '2011', y: 0.1, y1: 0.25, y2: 0.25, y3: 0, y4: 2 },
        { x: '2012', y: 0.25, y1: -0.5, y2: -0.1, y3: -0.4, y4: 0.4 },
        { x: '2013', y: 0.25, y1: 0.5, y2: -0.3, y3: 0, y4: 0.9 },
        { x: '2014', y: 0.6, y1: 0.6, y2: -0.6, y3: -0.6, y4: 0.4 },
        { x: '2015', y: 0.9, y1: 0.5, y2: 0, y3: -0.3, y4: 1.3 }
    ];
    const primaryxAxis: AxisModel = { title: 'Years', interval: 1,
labelIntersectAction: 'Rotate45', valueType: 'Category' };
    const primaryyAxis: AxisModel = { title: 'Growth', minimum: -3, maximum:
3, interval: 1 };
    const marker = { visible: true, width: 10, opacity: 0.6, height: 10 };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Annual Growth GDP in France'>
        <Inject services={[StackingColumnSeries, LineSeries, Legend, Tooltip,
DataLabel, Category]} />
        <SeriesCollectionDirective>

```

```

    <SeriesDirective dataSource={data} xName='x' yName='y' name='Private
Consumption'
      type='StackingColumn'>
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y1'
name='Government Consumption'
      type='StackingColumn'>
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y2'
name='Investment'
      type='StackingColumn'>
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y3' name='Net
Foreign Trade'
      type='StackingColumn'>
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y4' name='GDP'
type='Line' opacity={0.6}
      marker={marker}>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Note: Bar series cannot be combined with any other series as the axis orientation is different from other series.

<!-- markdownlint-disable MD036 -->

Technical indicators in React Chart component

A technical indicator is a mathematical calculation based on historic price, volume or open interest information that aims to forecast financial market direction.

Chart supports 10 types of technical indicators.

Accumulation Distribution

Accumulation Distribution combines price and volume to show how money may be flowing into or out of stock. To render a Accumulation Distribution Indicator, use indicator [type](#) as `AccumulationDistribution` and inject `AccumulationDistributionIndicator` into services. To calculate the signal line [volume](#) field is additionally added with `dataSource`.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, CandleSeries, Category, Tooltip,
DateTime, Logarithmic, Crosshair, LineSeries,
AccumulationDistributionIndicator, StripLine, IndicatorsDirective,
IndicatorDirective } from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const axisLabelRender = (args) => {

```

```

        if (args.axis.name === 'secondary') {
            let value = Number(args.text) / 1000000000;
            args.text = String(value) + 'bn';
        }
    };
    const primaryxAxis = { valueType: 'DateTime', intervalType: 'Months',
majorGridLines: { width: 0 }, crosshairTooltip: { enable: true } };
    const primaryyAxis = { title: 'Price', labelFormat: '${value}', minimum:
30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
    const animation = { enable: true };
    const chartarea = { border: { width: 0 } };
    const crosshair = { enable: true, lineType: 'Vertical' };
    const tooltip = { enable: true, shared: true };
    const lines = { width: 0 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} crosshair={crosshair}
axisLabelRender={axisLabelRender.bind(this)} chartArea={chartarea}
width={Browser.isDevice ? '100%' : '80%'} title='AAPL 2012-2017'>
        <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries,
AccumulationDistributionIndicator]}/>
        <AxesDirective>
            <AxisDirective rowIndex={0} name='secondary' opposedPosition={true}
majorGridLines={lines} majorTickLines={lines} minimum={-7000000000}
maximum={5000000000} interval={6000000000} title='Accumulation Distribution'
lineStyle={lines}>
                </AxisDirective>
            </AxesDirective>
            <SeriesCollectionDirective>
                <SeriesDirective dataSource={chartData} width={2} xName='x'
yName='y' low='low' high='high' close='close' volume='volume' open='open'
name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
type='Candle' animation={animation}>
                    </SeriesDirective>
                </SeriesCollectionDirective>
                <IndicatorsDirective>
                    <IndicatorDirective type='AccumulationDistribution' field='Close'
seriesName='Apple Inc' yAxisName='secondary' fill='#6063ff' period={3}
animation={animation}>
                        </IndicatorDirective>
                    </IndicatorsDirective>
                </ChartComponent>;
    }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { AxisModel, ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, RowDirective, RowsDirective, SeriesDirective, Inject,
CandleSeries, Category, Tooltip, ILoadedEventArgs, DateTime, Zoom,
Logarithmic, StripLineDirective, ChartAreaModel, CrosshairSettingsModel,

```

```

    Crosshair, LineSeries, AccumulationDistributionIndicator,
    IAxisLabelRenderEventArgs, TooltipSettingsModel,
    StripLine, ChartTheme, IndicatorsDirective, IndicatorDirective,
    StripLinesDirective}
from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const axisLabelRender = (args: IAxisLabelRenderEventArgs): void => {
        if (args.axis.name === 'secondary') {
            let value: number = Number(args.text) / 1000000000;
            args.text = String(value) + 'bn';
        }
    }
    const primaryxAxis: AxisModel = { valueType: 'DateTime', intervalType:
'Months', majorGridLines: { width: 0 }, crosshairTooltip: { enable: true }
};
    const primaryyAxis: AxisModel = { title: 'Price', labelFormat: '${value}',
minimum: 30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
    const animation = { enable: true };
    const chartarea: ChartAreaModel = { border: { width: 0 } };
    const crosshair: CrosshairSettingsModel = { enable: true, lineType:
'Vertical' };
    const tooltip: TooltipSettingsModel = { enable: true, shared: true };
    const lines = { width: 0 };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        tooltip={tooltip}
        crosshair={crosshair}
        axisLabelRender={axisLabelRender.bind(this)}
        chartArea={chartarea}
        width={Browser.isDevice ? '100%' : '80%'}
        title='AAPL 2012-2017'>
        <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries,
AccumulationDistributionIndicator]} />
        <AxesDirective>
            <AxisDirective rowIndex={0} name='secondary' opposedPosition={true}
                majorGridLines={lines} majorTickLines={lines} minimum={-
7000000000} maximum={5000000000}
                interval={6000000000} title='Accumulation Distribution'
                lineStyle={lines}>
            </AxisDirective>
        </AxesDirective>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} width={2}
                xName='x' yName='y' low='low' high='high' close='close'
                volume='volume' open='open'
                name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
                type='Candle' animation={animation}>
            </SeriesDirective>
        </SeriesCollectionDirective>
        <IndicatorsDirective>
            <IndicatorDirective type='AccumulationDistribution' field='Close'
                seriesName='Apple Inc' yAxisName='secondary' fill='#6063ff' period={3}
                animation={animation}>
            </IndicatorDirective>

```



```

    </IndicatorsDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Average True Range (ATR)

ATR measures the stock volatility by comparing the current value with the previous value.

To render a Average True Range (ATR) Indicator, use indicator [type](#) as `Atr` and inject `AtrIndicator` into services

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, CandleSeries, Category, Tooltip,
DateTime, Logarithmic, Crosshair, LineSeries, AtrIndicator, StripLine,
IndicatorsDirective, IndicatorDirective } from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'DateTime', intervalType: 'Months',
majorGridLines: { width: 0 }, crosshairTooltip: { enable: true } };
  const primaryyAxis = { title: 'Price', labelFormat: '${value}', minimum:
30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
  const animation = { enable: true };
  const chartarea = { border: { width: 0 } };
  const crosshair = { enable: true, lineType: 'Vertical' };
  const tooltip = { enable: true, shared: true };
  const lines = { width: 0 };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} crosshair={crosshair}
chartArea={chartarea} width={Browser.isDevice ? '100%' : '80%'} title='AAPL
2012-2017'>
    <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, AtrIndicator]} />
    <AxesDirective>
      <AxisDirective name='secondary' opposedPosition={true} rowIndex={0}
majorGridLines={lines} lineStyle={lines} majorTickLines={lines} maximum={14}
minimum={0} interval={7} title={'ATR'}>
      </AxisDirective>
    </AxesDirective>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} width={2} xName='x'
yName='y' low='low' high='high' close='close' volume='volume' open='open'
name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
type='Candle' animation={animation}>
      </SeriesDirective>
    </SeriesCollectionDirective>
    <IndicatorsDirective>
      <IndicatorDirective yAxisName='secondary' type='Atr' fill='#6063ff'
seriesName='Apple Inc' period={3} animation={animation}>
      </IndicatorDirective>

```

```

    </IndicatorsDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, RowDirective, RowsDirective, SeriesDirective, Inject,
CandleSeries, Category, Tooltip, ILoadedEventArgs, DateTime, Zoom,
Logarithmic, StripLinesDirective, StripLineDirective,
Crosshair, LineSeries, AtrIndicator, StripLine, ChartTheme,
IndicatorsDirective, IndicatorDirective
,AxisModel,ChartAreaModel,CrosshairSettingsModel, TooltipSettingsModel}
from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime', intervalType:
'Months', majorGridLines: { width: 0 }, crosshairTooltip: { enable: true }
};
  const primaryyAxis: AxisModel = { title: 'Price', labelFormat: '${value}',
minimum: 30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
  const animation = { enable: true };
  const chartarea: ChartAreaModel = { border: { width: 0 } };
  const crosshair: CrosshairSettingsModel = { enable: true, lineType:
'Vertical' };
  const tooltip: TooltipSettingsModel = { enable: true, shared: true };
  const lines = { width: 0 };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    tooltip={tooltip}
    crosshair={crosshair}
    chartArea={chartarea}
    width={Browser.isDevice ? '100%' : '80%'}
    title='AAPL 2012-2017'>
    <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, AtrIndicator]} />
    <AxesDirective>
      <AxisDirective name='secondary'
        opposedPosition={true} rowIndex={0}
        majorGridLines={lines} lineStyle={lines} majorTickLines={lines}
        maximum={14} minimum={0} interval={7} title={'ATR'}>
      </AxisDirective>
    </AxesDirective>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} width={2}
        xName='x' yName='y' low='low' high='high' close='close'
        volume='volume' open='open'
        name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
        type='Candle' animation={animation}>

```

```

    </SeriesDirective>
  </SeriesCollectionDirective>
  <IndicatorsDirective>
    <IndicatorDirective yAxisName='secondary' type='Atr' fill='#6063ff'
    seriesName='Apple Inc' period={3}
      animation={animation}>
    </IndicatorDirective>
  </IndicatorsDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Bollinger Band

A chart overlay that shows the upper and lower limits of normal price movements based on the standard deviation of prices.

To render a Bollinger Band, use indicator [type](#) as `BollingerBand` and inject `BollingerBands` module and `RangeAreaSeries` into services.

Bollinger band will be represented by three lines (upperLine, lowerLine, signalLine).

The default values of the Bollinger Band [period](#) is 14 and [standardDeviations](#) is 2.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
StripLine, CandleSeries, Category, Tooltip, DateTime, Logarithmic,
Crosshair, LineSeries, BollingerBands, IndicatorsDirective,
IndicatorDirective, RangeAreaSeries } from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'DateTime', intervalType: 'Months',
majorGridLines: { width: 0 }, crosshairTooltip: { enable: true } };
  const primaryyAxis = { title: 'Price', labelFormat: '${value}', minimum:
30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
  const animation = { enable: true };
  const chartarea = { border: { width: 0 } };
  const crosshair = { enable: true, lineType: 'Vertical' };
  const tooltip = { enable: true, shared: true };
  const upperline = { color: '#ffb735', width: 1 };
  const lowerline = { color: '#f2ec2f', width: 1 };
  const lines = { width: 0 };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} crosshair={crosshair}
chartArea={chartarea} width={Browser.isDevice ? '100%' : '80%'} title='AAPL
2012-2017'>
    <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, BollingerBands,
RangeAreaSeries]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} width={2} xName='x'
yName='y' low='low' high='high' close='close' volume='volume' open='open'

```

```

name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
type='Candle' animation={animation}>
  </SeriesDirective>
</SeriesCollectionDirective>
<IndicatorsDirective>
  <IndicatorDirective type='BollingerBands' field='Close'
seriesName='Apple Inc' fill='#606eff' period={14} animation={animation}
upperLine={upperline} lowerLine={lowerline}>
  </IndicatorDirective>
</IndicatorsDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
  ChartComponent, SeriesCollectionDirective, AxesDirective, AxisDirective,
  RowDirective, RowsDirective, SeriesDirective, Inject, StripLineDirective,
  StripLine, StripLinesDirective,
  CandleSeries, Category, Tooltip, ILoadedEventArgs, DateTime, Zoom,
  Logarithmic,
  Crosshair, LineSeries, BollingerBands, ChartTheme, IndicatorsDirective,
  IndicatorDirective, RangeAreaSeries,
  AxisModel, ChartAreaModel, CrosshairSettingsModel, TooltipSettingsModel
} from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime', intervalType:
'Months', majorGridLines: { width: 0 }, crosshairTooltip: { enable: true }
};
  const primaryyAxis: AxisModel = { title: 'Price', labelFormat: '${value}',
minimum: 30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
  const animation = { enable: true };
  const chartarea: ChartAreaModel = { border: { width: 0 } };
  const crosshair: CrosshairSettingsModel = { enable: true, lineType:
'Vertical' };
  const tooltip: TooltipSettingsModel = { enable: true, shared: true };
  const upperline = { color: '#ffb735', width: 1 };
  const lowerline = { color: '#f2ec2f', width: 1 };
  const lines = { width: 0 };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    tooltip={tooltip}
    crosshair={crosshair}
    chartArea={chartarea}
    width={Browser.isDevice ? '100%' : '80%'}
    title='AAPL 2012-2017'>

```

```

    <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, BollingerBands,
RangeAreaSeries]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} width={2}
        xName='x' yName='y' low='low' high='high' close='close'
        volume='volume' open='open'
        name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
        type='Candle' animation={animation}>
      </SeriesDirective>
    </SeriesCollectionDirective>
    <IndicatorsDirective>
      <IndicatorDirective type='BollingerBands' field='Close'
        seriesName='Apple Inc' fill='#606eff'
        period={14} animation={animation} upperLine={upperline}
        lowerLine={lowerline}>
      </IndicatorDirective>
    </IndicatorsDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Customization of BollingerBand

stroke, stroke-width, and color of upperLine can be customized by using [upperLine](#), and the lowerLine can be customized by using [lowerLine](#) properties of indicator.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
StripLine, CandleSeries, Category, Tooltip, DateTime, Logarithmic,
Crosshair, LineSeries, BollingerBands, IndicatorsDirective,
IndicatorDirective } from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'DateTime', intervalType: 'Months',
majorGridLines: { width: 0 }, crosshairTooltip: { enable: true } };
  const primaryyAxis = { title: 'Price', labelFormat: '${value}', minimum:
30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
  const animation = { enable: true };
  const chartarea = { border: { width: 0 } };
  const crosshair = { enable: true, lineType: 'Vertical' };
  const tooltip = { enable: true, shared: true };
  const upperline = { color: 'orange' };
  const lowerline = { color: 'yellow' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} crosshair={crosshair}
chartArea={chartarea} width={Browser.isDevice ? '100%' : '80%'} title='AAPL
2012-2017'>
    <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, BollingerBands]} />
    <SeriesCollectionDirective>

```

```

    <SeriesDirective dataSource={chartData} width={2} xName='x'
    yName='y' low='low' high='high' close='close' volume='volume' open='open'
    name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
    type='Candle' animation={animation}>
    </SeriesDirective>
  </SeriesCollectionDirective>
  <IndicatorsDirective>
    <IndicatorDirective type='BollingerBands' field='Close'
    seriesName='Apple Inc' fill='#606eff' period={14} animation={animation}
    upperLine={upperline} lowerLine={lowerline}>
    </IndicatorDirective>
  </IndicatorsDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
  ChartComponent, SeriesCollectionDirective, AxesDirective, AxisDirective,
  RowDirective, RowsDirective, SeriesDirective, Inject, StripLineDirective,
  StripLine, StripLinesDirective,
  CandleSeries, Category, Tooltip, ILoadedEventArgs, DateTime, Zoom,
  Logarithmic,
  Crosshair, LineSeries, BollingerBands, ChartTheme, IndicatorsDirective,
  IndicatorDirective, RangeAreaSeries,
  AxisModel, ChartAreaModel, CrosshairSettingsModel, TooltipSettingsModel
} from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime', intervalType:
'Months', majorGridLines: { width: 0 }, crosshairTooltip: { enable: true }
};
  const primaryyAxis: AxisModel = { title: 'Price', labelFormat: '${value}',
minimum: 30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
  const animation = { enable: true };
  const chartarea: ChartAreaModel = { border: { width: 0 } };
  const crosshair: CrosshairSettingsModel = { enable: true, lineType:
'Vertical' };
  const tooltip: TooltipSettingsModel = { enable: true, shared: true };
  const upperline = { color: 'orange' };
  const lowerline = { color: 'yellow' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    tooltip={tooltip}
    crosshair={crosshair}
    chartArea={chartarea}
    width={Browser.isDevice ? '100%' : '80%'}
    title='AAPL 2012-2017'>

```

```

    <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, BollingerBands]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} width={2}
        xName='x' yName='y' low='low' high='high' close='close'
volume='volume' open='open'
        name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
        type='Candle' animation={animation}>
      </SeriesDirective>
    </SeriesCollectionDirective>
    <IndicatorsDirective>
      <IndicatorDirective type='BollingerBands' field='Close'
seriesName='Apple Inc' fill='#606eff'
        period={14} animation={animation} upperLine={upperline}
lowerLine={lowerline}>
      </IndicatorDirective>
    </IndicatorsDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Exponential Moving Average (EMA)

Moving average Indicators are used to define the direction of the trend. To render a EMA Indicator, use indicator [type](#) as `Ema` and inject `EmaIndicator` into services.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
StripLine, CandleSeries, Category, Tooltip, DateTime, Logarithmic,
Crosshair, LineSeries, EmaIndicator, IndicatorsDirective, IndicatorDirective
} from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'DateTime', intervalType: 'Months',
majorGridLines: { width: 0 }, crosshairTooltip: { enable: true } };
  const primaryyAxis = { title: 'Price', labelFormat: '${value}', minimum:
30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
  const animation = { enable: true };
  const chartarea = { border: { width: 0 } };
  const crosshair = { enable: true, lineType: 'Vertical' };
  const tooltip = { enable: true, shared: true };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} crosshair={crosshair}
chartArea={chartarea} width={Browser.isDevice ? '100%' : '80%'} title='AAPL
2012-2017'>
    <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, EmaIndicator]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} width={2} xName='x'
yName='y' low='low' high='high' close='close' volume='volume' open='open'

```

```

name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
type='Candle' animation={animation}>
  </SeriesDirective>
</SeriesCollectionDirective>
<IndicatorsDirective>
  <IndicatorDirective type='Ema' field='Close' seriesName='Apple Inc'
fill='#606eff' period={14} animation={animation}>
  </IndicatorDirective>
</IndicatorsDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
  ChartComponent, SeriesCollectionDirective, AxesDirective, AxisDirective,
  RowDirective, RowsDirective, SeriesDirective, Inject, StripLineDirective,
  StripLine, StripLinesDirective,
  CandleSeries, Category, Tooltip, ILoadedEventArgs, DateTime, Zoom,
  Logarithmic, ChartTheme,
  Crosshair, LineSeries, EmaIndicator, IndicatorsDirective,
  IndicatorDirective, AxisModel, ChartAreaModel, CrosshairSettingsModel,
  TooltipSettingsModel
} from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryXAxis: AxisModel = { valueType: 'DateTime', intervalType:
'Months', majorGridLines: { width: 0 }, crosshairTooltip: { enable: true }
};
  const primaryYAxis: AxisModel = { title: 'Price', labelFormat: '${value}',
minimum: 30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
  const animation = { enable: true };
  const chartarea: ChartAreaModel = { border: { width: 0 } };
  const crosshair: CrosshairSettingsModel = { enable: true, lineType:
'Vertical' };
  const tooltip: TooltipSettingsModel = { enable: true, shared: true };
  return <ChartComponent id='charts'
    primaryXAxis={primaryXAxis}
    primaryYAxis={primaryYAxis}
    tooltip={tooltip}
    crosshair={crosshair}
    chartArea={chartarea}
    width={Browser.isDevice ? '100%' : '80%'}
    title='AAPL 2012-2017'>
    <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, EmaIndicator]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} width={2}
        xName='x' yName='y' low='low' high='high' close='close'
        volume='volume' open='open'

```



```

        name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
type='Candle' animation={animation}>
    </SeriesDirective>
</SeriesCollectionDirective>
<IndicatorsDirective>
    <IndicatorDirective type='Ema' field='Close' seriesName='Apple Inc'
fill='#606eff' period={14} animation={animation}>
    </IndicatorDirective>
</IndicatorsDirective>
</ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Momentum

Momentum shows the speed at which the price of the stock is changing. To render a Momentum indicator, use indicator `type` as `Momentum` and inject `MomentumIndicator` module into services. Momentum indicator will be represented by two lines (upperLine, signalLine). In momentum indicator the upperBand value is always renders at the value 100.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, CandleSeries, Category, Tooltip,
DateTime, Logarithmic, Crosshair, LineSeries, MomentumIndicator, StripLine,
IndicatorsDirective, IndicatorDirective } from '@syncfusion/ej2-react-
charts';
import { chartData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'DateTime', intervalType: 'Months',
majorGridLines: { width: 0 }, crosshairTooltip: { enable: true } };
    const primaryyAxis = { title: 'Price', labelFormat: '${value}', minimum:
30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
    const animation = { enable: true };
    const chartarea = { border: { width: 0 } };
    const crosshair = { enable: true, lineType: 'Vertical' };
    const tooltip = { enable: true, shared: true };
    const lines = { width: 0 };
    const upperline = { color: '#e74c3d' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} crosshair={crosshair}
chartArea={chartarea} width={Browser.isDevice ? '100%' : '80%'} title='AAPL
2012-2017'>
        <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, MomentumIndicator,
StripLine]} />
        <AxesDirective>
            <AxisDirective rowIndex={0} name='secondary' opposedPosition={true}
majorGridLines={lines} majorTickLines={lines} minimum={80} maximum={120}
interval={20} title='Momentum' plotOffset={30} lineStyle={lines}>
            </AxisDirective>

```

```

    </AxesDirective>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} width={2} xName='x'
yName='y' low='low' high='high' close='close' volume='volume' open='open'
name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
type='Candle' animation={animation}>
    </SeriesDirective>
  </SeriesCollectionDirective>
  <IndicatorsDirective>
    <IndicatorDirective type='Momentum' field='Close' seriesName='Apple
Inc' yAxisName='secondary' fill='#6063ff' period={3} animation={animation}
upperLine={upperline}>
    </IndicatorDirective>
  </IndicatorsDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
  ChartComponent, SeriesCollectionDirective, AxesDirective, AxisDirective,
  RowDirective, RowsDirective, SeriesDirective, Inject,
  CandleSeries, Category, Tooltip, ILoadedEventArgs, DateTime, Zoom,
  Logarithmic, StripLinesDirective, StripLineDirective,
  Crosshair, LineSeries, MomentumIndicator, StripLine, ChartTheme,
  IndicatorsDirective, IndicatorDirective,
  AxisModel, ChartAreaModel, CrosshairSettingsModel, TooltipSettingsModel
} from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime', intervalType:
'Months', majorGridLines: { width: 0 }, crosshairTooltip: { enable: true }
};
  const primaryyAxis: AxisModel = { title: 'Price', labelFormat: '${value}',
minimum: 30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
  const animation = { enable: true };
  const chartarea: ChartAreaModel = { border: { width: 0 } };
  const crosshair: CrosshairSettingsModel = { enable: true, lineType:
'Vertical' };
  const tooltip: TooltipSettingsModel = { enable: true, shared: true };
  const lines = { width: 0 };
  const upperline = { color: '#e74c3d' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    tooltip={tooltip}
    crosshair={crosshair}
    chartArea={chartarea}
    width={Browser.isDevice ? '100%' : '80%'}
    title='AAPL 2012-2017'>

```

```

    <Inject services={[CandleSeries, Category, Tooltip, StripLine,
    DateTime, Logarithmic, Crosshair, LineSeries, MomentumIndicator, StripLine]}
    />

    <AxesDirective>
      <AxisDirective rowIndex={0} name='secondary' opposedPosition={true}
      majorGridLines={lines} majorTickLines={lines} minimum={80} maximum={120}
      interval={20} title='Momentum' plotOffset={30} lineStyle={lines}>
        </AxisDirective>
      </AxesDirective>
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={chartData} width={2}
        xName='x' yName='y' low='low' high='high' close='close'
        volume='volume' open='open'
        name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
        type='Candle' animation={animation}>
          </SeriesDirective>
        </SeriesCollectionDirective>
        <IndicatorsDirective>
          <IndicatorDirective type='Momentum' field='Close' seriesName='Apple
          Inc' yAxisName='secondary' fill='#6063ff'
          period={3} animation={animation} upperLine={upperline}>
            </IndicatorDirective>
          </IndicatorsDirective>
        </ChartComponent>
      };
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));

```

Customization of MomentumIndicator

stroke, stroke-width, and color of upperLine can be customized by using [upperLine](#) property of indicator.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, CandleSeries, Category, Tooltip,
DateTime, Logarithmic, Crosshair, LineSeries, MomentumIndicator, StripLine,
IndicatorsDirective, IndicatorDirective } from '@syncfusion/ej2-react-
charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'DateTime', intervalType: 'Months',
  majorGridLines: { width: 0 }, crosshairTooltip: { enable: true } };
  const primaryyAxis = { title: 'Price', labelFormat: '${value}', minimum:
  30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
  const animation = { enable: true };
  const chartarea = { border: { width: 0 } };
  const crosshair = { enable: true, lineType: 'Vertical' };
  const tooltip = { enable: true, shared: true };
  const lines = { width: 0 };
  const upperline = { color: '#e74c3d' };

```

```

    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis} tooltip={tooltip} crosshair={crosshair}
    chartArea={chartarea} width={Browser.isDevice ? '100%' : '80%'} title='AAPL
    2012-2017'>
      <Inject services={[CandleSeries, Category, Tooltip, StripLine,
    DateTime, Logarithmic, Crosshair, LineSeries, MomentumIndicator,
    StripLine]}/>
      <AxesDirective>
        <AxisDirective rowIndex={0} name='secondary' opposedPosition={true}
    majorGridLines={lines} majorTickLines={lines} minimum={80} maximum={120}
    interval={20} title='Momentum' plotOffset={30} lineStyle={lines}>
          </AxisDirective>
        </AxesDirective>
        <SeriesCollectionDirective>
          <SeriesDirective dataSource={chartData} width={2} xName='x'
    yName='y' low='low' high='high' close='close' volume='volume' open='open'
    name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
    type='Candle' animation={animation}>
            </SeriesDirective>
          </SeriesCollectionDirective>
          <IndicatorsDirective>
            <IndicatorDirective type='Momentum' field='Close' seriesName='Apple
    Inc' yAxisName='secondary' fill='#6063ff' period={3} animation={animation}
    upperLine={upperline}>
              </IndicatorDirective>
            </IndicatorsDirective>
          </ChartComponent>;
    }
    ;
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
    ChartComponent, SeriesCollectionDirective, AxesDirective, AxisDirective,
    RowDirective, RowsDirective, SeriesDirective, Inject,
    CandleSeries, Category, Tooltip, ILoadedEventArgs, DateTime, Zoom,
    Logarithmic, StripLinesDirective, StripLineDirective,
    Crosshair, LineSeries, MomentumIndicator, StripLine, ChartTheme,
    IndicatorsDirective, IndicatorDirective,
    AxisModel, ChartAreaModel, CrosshairSettingsModel, TooltipSettingsModel
} from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime', intervalType:
    'Months', majorGridLines: { width: 0 }, crosshairTooltip: { enable: true }
    };
    const primaryyAxis: AxisModel = { title: 'Price', labelFormat: '${value}',
    minimum: 30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
    const animation = { enable: true };
    const chartarea: ChartAreaModel = { border: { width: 0 } };

```

```

const crosshair: CrosshairSettingsModel = { enable: true, lineType:
'Vertical' };
const tooltip: TooltipSettingsModel = { enable: true, shared: true };
const lines = { width: 0 };
const upperline = { color: '#e74c3d' };
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  tooltip={tooltip}
  crosshair={crosshair}
  chartArea={chartarea}
  width={Browser.isDevice ? '100%' : '80%'}
  title='AAPL 2012-2017'>
  <Inject services=[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, MomentumIndicator, StripLine]]
/>
  <AxesDirective>
    <AxisDirective rowIndex={0} name='secondary' opposedPosition={true}
majorGridLines={lines} majorTickLines={lines} minimum={80} maximum={120}
interval={20} title='Momentum' plotOffset={30} lineStyle={lines}>
    </AxisDirective>
  </AxesDirective>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={chartData} width={2}
      xName='x' yName='y' low='low' high='high' close='close'
volume='volume' open='open'
      name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
      type='Candle' animation={animation}>
    </SeriesDirective>
  </SeriesCollectionDirective>
  <IndicatorsDirective>
    <IndicatorDirective type='Momentum' field='Close' seriesName='Apple
Inc' yAxisName='secondary' fill='#6063ff'
      period={3} animation={animation} upperLine={upperline}>
    </IndicatorDirective>
  </IndicatorsDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Moving Average Convergence Divergence (MACD)

MACD is based on the difference between two EMA's. To render a MACD Indicator, use indicator [type](#) as `Macd` and inject `MacdIndicator` module using into services. MACD indicator will be represented by MACD line, signal line, MACD histogram. MACD histogram is used to differentiate MACD line and signal line.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, CandleSeries, Category, Tooltip,
DateTime, Logarithmic, Crosshair, LineSeries, StripLine, MacdIndicator,

```

```

ColumnSeries, IndicatorsDirective, IndicatorDirective } from
'@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'DateTime', intervalType: 'Months',
majorGridLines: { width: 0 }, crosshairTooltip: { enable: true } };
    const primaryyAxis = { title: 'Price', labelFormat: '${value}', minimum:
30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
    const animation = { enable: true };
    const chartarea = { border: { width: 0 } };
    const crosshair = { enable: true, lineType: 'Vertical' };
    const tooltip = { enable: true, shared: true };
    const lines = { width: 0 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} crosshair={crosshair}
chartArea={chartarea} width={Browser.isDevice ? '100%' : '80%'} title='AAPL
2012-2017'>
        <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, MacdIndicator, StripLine,
ColumnSeries]}/>
        <AxesDirective>
            <AxisDirective opposedPosition={true} rowIndex={0} name='secondary'
majorGridLines={lines} lineStyle={lines} minimum={-3.5} maximum={3.5}
interval={3.5} majorTickLines={lines} title='MACD'>
            </AxisDirective>
        </AxesDirective>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} width={2} xName='x'
yName='y' low='low' high='high' close='close' volume='volume' open='open'
name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
type='Candle' animation={animation}>
            </SeriesDirective>
        </SeriesCollectionDirective>
        <IndicatorsDirective>
            <IndicatorDirective type='Macd' period={3} fastPeriod={8}
slowPeriod={5} seriesName='Apple Inc' macdType='Both' width={2}
macdPositiveColor='#2ecd71' macdNegativeColor='#e74c3d' fill='#6063ff'
yAxisName='secondary'>
            </IndicatorDirective>
        </IndicatorsDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
    ChartComponent, SeriesCollectionDirective, AxesDirective, AxisDirective,
    RowDirective, RowsDirective, SeriesDirective, Inject,
    CandleSeries, Category, Tooltip, ILoadedEventArgs, DateTime, Zoom,
    Logarithmic, StripLinesDirective, StripLineDirective,

```

```

    Crosshair, LineSeries, StripLine, MacdIndicator, ColumnSeries,
    ChartTheme, IndicatorsDirective, IndicatorDirective,
    AxisModel, ChartAreaModel, CrosshairSettingsModel, TooltipSettingsModel
} from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime', intervalType:
'Months', majorGridLines: { width: 0 }, crosshairTooltip: { enable: true }
};
    const primaryyAxis: AxisModel = { title: 'Price', labelFormat: '${value}',
minimum: 30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
    const animation = { enable: true };
    const chartarea: ChartAreaModel = { border: { width: 0 } };
    const crosshair: CrosshairSettingsModel = { enable: true, lineType:
'Vertical' };
    const tooltip: TooltipSettingsModel = { enable: true, shared: true };
    const lines = { width: 0 };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        tooltip={tooltip}
        crosshair={crosshair}
        chartArea={chartarea}
        width={Browser.isDevice ? '100%' : '80%'}
        title='AAPL 2012-2017'>
        <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, MacdIndicator, StripLine,
ColumnSeries]} />
        <AxesDirective>
            <AxisDirective opposedPosition={true} rowIndex={0} name='secondary'
                majorGridLines={lines} lineStyle={lines} minimum={-3.5}
maximum={3.5} interval={3.5}
                majorTickLines={lines} title='MACD'>
            </AxisDirective>
        </AxesDirective>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} width={2}
                xName='x' yName='y' low='low' high='high' close='close'
volume='volume' open='open'
                name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
                type='Candle' animation={animation}>
            </SeriesDirective>
        </SeriesCollectionDirective>
        <IndicatorsDirective>
            <IndicatorDirective type='Macd' period={3} fastPeriod={8}
slowPeriod={5} seriesName='Apple Inc' macdType='Both' width={2}
macdPositiveColor='#2ecd71' macdNegativeColor='#e74c3d' fill='#6063ff'
yAxisName='secondary'>
            </IndicatorDirective>
        </IndicatorsDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Customization of MACD

`stroke`, `stroke-width`, and `color` of `macdLine` can be customized by using, [macdLine](#), property of indicator. The positive and negative changes of histogram can be customized by [macdPositiveColor](#) and [macdNegativeColor](#) properties. The `[macdType]` is used to define the type of MACD indicator. To customize the MACD period using [slowPeriod](#) and [fastPeriod](#) properties.

By default `macdType` as 'Both'.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, CandleSeries, Category, Tooltip,
DateTime, Logarithmic, Crosshair, LineSeries, StripLine, MacdIndicator,
ColumnSeries, IndicatorsDirective, IndicatorDirective } from
 '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'DateTime', intervalType: 'Months',
majorGridLines: { width: 0 }, crosshairTooltip: { enable: true } };
    const primaryyAxis = { title: 'Price', labelFormat: '${value}', minimum:
30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
    const animation = { enable: true };
    const chartarea = { border: { width: 0 } };
    const crosshair = { enable: true, lineType: 'Vertical' };
    const tooltip = { enable: true, shared: true };
    const lines = { width: 0 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} crosshair={crosshair}
chartArea={chartarea} width={Browser.isDevice ? '100%' : '80%'} title='AAPL
2012-2017'>
        <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, MacdIndicator, StripLine,
ColumnSeries]}/>
        <AxesDirective>
            <AxisDirective opposedPosition={true} rowIndex={0} name='secondary'
majorGridLines={lines} lineStyle={lines} minimum={-3.5} maximum={3.5}
interval={3.5} majorTickLines={lines} title='MACD'>
                </AxisDirective>
            </AxesDirective>
            <SeriesCollectionDirective>
                <SeriesDirective dataSource={chartData} width={2} xName='x'
yName='y' low='low' high='high' close='close' volume='volume' open='open'
name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
type='Candle' animation={animation}>
                    </SeriesDirective>
                </SeriesCollectionDirective>
                <IndicatorsDirective>
                    <IndicatorDirective type='Macd' period={3} fastPeriod={8}
slowPeriod={5} seriesName='Apple Inc' macdType='Both' width={2}
macdPositiveColor='#2ecd71' macdNegativeColor='#e74c3d' fill='#6063ff'
yAxisName='secondary'>
                        </IndicatorDirective>
                    </IndicatorsDirective>
                </ChartComponent>;
    }
```



```
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
    ChartComponent, SeriesCollectionDirective, AxesDirective, AxisDirective,
    RowDirective, RowsDirective, SeriesDirective, Inject,
    CandleSeries, Category, Tooltip, ILoadedEventArgs, DateTime, Zoom,
    Logarithmic, StripLinesDirective, StripLineDirective,
    Crosshair, LineSeries, StripLine, MacdIndicator, ColumnSeries,
    ChartTheme, IndicatorsDirective, IndicatorDirective,
    AxisModel, ChartAreaModel, CrosshairSettingsModel, TooltipSettingsModel
} from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime', intervalType:
'Months', majorGridLines: { width: 0 }, crosshairTooltip: { enable: true }
};
    const primaryyAxis: AxisModel = { title: 'Price', labelFormat: '${value}',
minimum: 30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
    const animation = { enable: true };
    const chartarea: ChartAreaModel = { border: { width: 0 } };
    const crosshair: CrosshairSettingsModel = { enable: true, lineType:
'Vertical' };
    const tooltip: TooltipSettingsModel = { enable: true, shared: true };
    const lines = { width: 0 };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        tooltip={tooltip}
        crosshair={crosshair}
        chartArea={chartarea}
        width={Browser.isDevice ? '100%' : '80%'}
        title='AAPL 2012-2017'>
        <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, MacdIndicator, StripLine,
ColumnSeries]} />
        <AxesDirective>
            <AxisDirective opposedPosition={true} rowIndex={0} name='secondary'
                majorGridLines={lines} lineStyle={lines} minimum={-3.5}
maximum={3.5} interval={3.5}
                majorTickLines={lines} title='MACD'>
            </AxisDirective>
        </AxesDirective>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} width={2}
                xName='x' yName='y' low='low' high='high' close='close'
volume='volume' open='open'
                name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
                type='Candle' animation={animation}>
            </SeriesDirective>
```

```

    </SeriesCollectionDirective>
    <IndicatorsDirective>
      <IndicatorDirective type='Macd' period={3} fastPeriod={8}
slowPeriod={5} seriesName='Apple Inc' macdType='Both' width={2}
macdPositiveColor='#2ecd71' macdNegativeColor='#e74c3d' fill='#6063ff'
yAxisName='secondary'>
    </IndicatorDirective>
  </IndicatorsDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Relative Strength Index (RSI)

RSI shows how strongly a stock is moving in its current direction. To render a RSI Indicator, use indicator [type](#) as `Rsi` and inject `RsiIndicator` module into services. RSI indicator will be represented by three lines (upperBand, lowerBand, signalLine). The upperBand and lowerBand values are customized by [overBought](#) and [overSold](#) properties of indicator and the signalLine is calculated by RSI formula.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, LineSeries, RsiIndicator,
IndicatorsDirective, IndicatorDirective, Category, StripLine, CandleSeries,
Tooltip, DateTime, Logarithmic, Crosshair } from '@syncfusion/ej2-react-
charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryXAxis = { valueType: 'DateTime', intervalType: 'Months',
majorGridLines: { width: 0 }, crosshairTooltip: { enable: true } };
  const primaryYAxis = { title: 'Price', labelFormat: '$ {value}', minimum:
30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
  const animation = { enable: true };
  const chartarea = { border: { width: 0 } };
  const crosshair = { enable: true, lineType: 'Vertical' };
  const tooltip = { enable: true, shared: true };
  const lines = { width: 0 };
  const upperline = { color: '#e74c3d' };
  const lowerline = { color: '#2ecd71' };
  return <ChartComponent id='charts' primaryXAxis={primaryXAxis}
primaryYAxis={primaryYAxis} tooltip={tooltip} crosshair={crosshair}
chartArea={chartarea} width={Browser.isDevice ? '100%' : '80%'} title='AAPL
2012-2017'>
    <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, RsiIndicator, StripLine]} />
    <AxesDirective>
      <AxisDirective rowIndex={0} name='secondary' opposedPosition={true}
majorGridLines={lines} majorTickLines={lines} minimum={0} maximum={120}
interval={60} title='RSI' lineStyle={lines}>
    </AxisDirective>
  </AxesDirective>
  <SeriesCollectionDirective>

```

```

    <SeriesDirective dataSource={chartData} xName='x' yName='silver'
    name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d' low='low'
    open='open' close='close' high='high' volume='volume' type='Candle'>
    </SeriesDirective>
  </SeriesCollectionDirective>
  <IndicatorsDirective>
    <IndicatorDirective field='Close' yAxisName='secondary' type='Rsi'
    fill='#6063ff' seriesName='Apple Inc' period={3} showZones={true}
    overBought={70} overSold={30} upperLine={upperline} lowerLine={lowerline}
    animation={animation}>
    </IndicatorDirective>
  </IndicatorsDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
  ChartComponent, SeriesCollectionDirective, AxesDirective, AxisDirective,
  RowDirective, RowsDirective,
  SeriesDirective, Inject, ILoadedEventArgs, StripLinesDirective,
  StripLineDirective,
  LineSeries, RsiIndicator, IndicatorsDirective, ChartTheme,
  IndicatorDirective, Category, StripLine, CandleSeries, Tooltip,
  DateTime, Zoom, Logarithmic, Crosshair,
  AxisModel, ChartAreaModel, CrosshairSettingsModel, TooltipSettingsModel
} from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime', intervalType:
'Months', majorGridLines: { width: 0 }, crosshairTooltip: { enable: true }
};
  const primaryyAxis: AxisModel = { title: 'Price', labelFormat: '$ {value}',
minimum: 30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
  const animation = { enable: true };
  const chartarea: ChartAreaModel = { border: { width: 0 } };
  const crosshair: CrosshairSettingsModel = { enable: true, lineType:
'Vertical' };
  const tooltip: TooltipSettingsModel = { enable: true, shared: true };
  const lines = { width: 0 };
  const upperline = { color: '#e74c3d' };
  const lowerline = { color: '#2ecd71' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    tooltip={tooltip}
    crosshair={crosshair}
    chartArea={chartarea}
    width={Browser.isDevice ? '100%' : '80%'}
    title='AAPL 2012-2017'>

```

```

    <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, RsiIndicator, StripLine]} />
    <AxesDirective>
      <AxisDirective rowIndex={0} name='secondary' opposedPosition={true}
majorGridLines={lines}
      majorTickLines={lines} minimum={0} maximum={120} interval={60}
title='RSI' lineStyle={lines}>
    </AxisDirective>
  </AxesDirective>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={chartData} xName='x' yName='silver'
name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d' low='low'
open='open' close='close' high='high' volume='volume' type='Candle'>
    </SeriesDirective>
  </SeriesCollectionDirective>
  <IndicatorsDirective>
    <IndicatorDirective field='Close' yAxisName='secondary' type='Rsi'
fill='#6063ff' seriesName='Apple Inc' period={3}
    showZones={true} overBought={70} overSold={30}
upperLine={upperline} lowerLine={lowerline}
    animation={animation}>
    </IndicatorDirective>
  </IndicatorsDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Simple Moving Average (SMA)

Moving average Indicators are used to define the direction of the trend. To render a SMA Indicator, use indicator [type](#) as `Sma` and inject `SmaIndicator` module into services.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
DateTime, Logarithmic, Tooltip, CandleSeries, Crosshair, LineSeries,
SmaIndicator, IndicatorsDirective, IndicatorDirective, StripLine } from
 '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'DateTime', intervalType: 'Months',
majorGridLines: { width: 0 }, crosshairTooltip: { enable: true } };
  const primaryyAxis = {
    title: 'Price', labelFormat: '${value}', minimum: 30, maximum: 180,
interval: 30,
    lineStyle: { width: 0 }
  };
  const animation = { enable: true };
  const chartarea = { border: { width: 0 } };
  const crosshair = { enable: true, lineType: 'Vertical' };
  const tooltip = { enable: true, shared: true };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} crosshair={crosshair}

```

```

chartArea={chartarea} width={Browser.isDevice ? '100%' : '80%'} title='AAPL
2012-2017'>
  <Inject services={[CandleSeries, Tooltip, StripLine, DateTime,
Logarithmic, Crosshair, LineSeries, SmaIndicator]}/>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={chartData} width={2} xName='x'
yName='silver' low='low' high='high' close='close' volume='volume'
open='open' name='Apple Inc' type='Candle' animation={animation}>
    </SeriesDirective>
  </SeriesCollectionDirective>
  <IndicatorsDirective>
    <IndicatorDirective type='Sma' fill='blue' seriesName='Apple Inc'
period={14}>
    </IndicatorDirective>
  </IndicatorsDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
  ChartComponent, SeriesCollectionDirective, AxesDirective, AxisDirective,
  RowDirective, RowsDirective, SeriesDirective, Inject, Legend, DateTime,
  Logarithmic, Tooltip, CandleSeries, DataLabel, Crosshair, Zoom,
  ColumnSeries, ILoadedEventArgs,
  LineSeries, SmaIndicator, IndicatorsDirective, IndicatorDirective,
  ChartTheme, StripLineDirective, StripLine, StripLinesDirective,
  AxisModel, ChartAreaModel, CrosshairSettingsModel, TooltipSettingsModel
} from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime', intervalType:
'Months', majorGridLines: { width: 0 }, crosshairTooltip: { enable: true }
};
  const primaryyAxis: AxisModel = {
    title: 'Price', labelFormat: '${value}', minimum: 30, maximum: 180,
    interval: 30,
    lineStyle: { width: 0 }
  };
  const animation = { enable: true };
  const chartarea: ChartAreaModel = { border: { width: 0 } };
  const crosshair: CrosshairSettingsModel = { enable: true, lineType:
'Vertical' };
  const tooltip: TooltipSettingsModel = { enable: true, shared: true };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    tooltip={tooltip}
    crosshair={crosshair}
    chartArea={chartarea}

```

```

width={Browser.isDevice ? '100%' : '80%'}
title='AAPL 2012-2017'>
<Inject services={[CandleSeries, Tooltip, StripLine, DateTime,
Logarithmic, Crosshair, LineSeries, SmaIndicator]} />
<SeriesCollectionDirective>
  <SeriesDirective dataSource={chartData} width={2}
    xName='x' yName='silver' low='low' high='high' close='close'
    volume='volume' open='open'
    name='Apple Inc' type='Candle' animation={animation}>
  </SeriesDirective>
</SeriesCollectionDirective>
<IndicatorsDirective>
  <IndicatorDirective type='Sma' fill='blue' seriesName='Apple Inc'
period={14}>
  </IndicatorDirective>
</IndicatorsDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Stochastic

It shows how a stock is, when compared to previous state. To render a Stochastic indicator, use indicator [type](#) as **Stochastic** and inject **StochasticIndicator** module into services.

stochastic indicator will be represented by four lines (upperLine, lowerLine, periodLine, signalLine).

In stochastic indicator the upperBand value and lowerBand value is customized by [overBought](#) and [overSold](#) properties of indicators and the periodLine and signalLine is render based on stochastic formula.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, StripLine, DateTime, Logarithmic,
Tooltip, CandleSeries, Crosshair, LineSeries, StochasticIndicator,
IndicatorsDirective, IndicatorDirective, Category } from '@syncfusion/ej2-
react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'DateTime', intervalType: 'Months',
majorGridLines: { width: 0 }, crosshairTooltip: { enable: true } };
  const primaryyAxis = { title: 'Price', labelFormat: '${value}', minimum:
30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
  const animation = { enable: true };
  const chartarea = { border: { width: 0 } };
  const crosshair = { enable: true, lineType: 'Vertical' };
  const tooltip = { enable: true, shared: true };
  const lines = { width: 0 };
  const animation1 = { enable: false };
  const upperline = { color: '#e74c3d' };
  const lowerline = { color: '#2ecc71' };
  const periodline = { color: '#f2ec2f' };

```

```

    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} crosshair={crosshair}
chartArea={chartarea} width={Browser.isDevice ? '100%' : '80%'} title='AAPL
2012-2017'>
    <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, StochasticIndicator]}/>
    <AxesDirective>
        <AxisDirective rowIndex={0} name='secondary' opposedPosition={true}
majorGridLines={lines} majorTickLines={lines} minimum={0} maximum={120}
interval={60} title='Stochastic' lineStyle={lines}>
        </AxisDirective>
    </AxesDirective>
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={chartData} width={2} xName='x'
yName='y' low='low' high='high' close='close' volume='volume' open='open'
name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
type='Candle' animation={animation}>
        </SeriesDirective>
    </SeriesCollectionDirective>
    <IndicatorsDirective>
        <IndicatorDirective type='Stochastic' field='Close'
seriesName='Apple Inc' yAxisName='secondary' fill='#6063ff' kPeriod={2}
dPeriod={3} showZones={true} periodLine={periodline} period={3}
animation={animation1} upperLine={upperline} lowerLine={lowerline}>
        </IndicatorDirective>
    </IndicatorsDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
    ChartComponent, SeriesCollectionDirective, AxesDirective, AxisDirective,
    RowDirective, RowsDirective, SeriesDirective, Inject,
    StripLine, DateTime, Logarithmic, Tooltip, CandleSeries, Crosshair,
    Zoom, ILoadedEventArgs, StripLinesDirective, StripLineDirective,
    LineSeries, StochasticIndicator, IndicatorsDirective,
    IndicatorDirective, Category, ChartTheme,
    AxisModel, ChartAreaModel, CrosshairSettingsModel, TooltipSettingsModel
} from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime', intervalType:
'Months', majorGridLines: { width: 0 }, crosshairTooltip: { enable: true }
};
    const primaryyAxis: AxisModel = { title: 'Price', labelFormat: '${value}',
minimum: 30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
    const animation = { enable: true };
    const chartarea: ChartAreaModel = { border: { width: 0 } };

```

```

const crosshair: CrosshairSettingsModel = { enable: true, lineType:
'Vertical' };
const tooltip: TooltipSettingsModel = { enable: true, shared: true };
const lines = { width: 0 };
const animation1 = { enable: false };
const upperline = { color: '#e74c3d' };
const lowerline = { color: '#2ecd71' };
const periodline = { color: '#f2ec2f' };
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  tooltip={tooltip}
  crosshair={crosshair}
  chartArea={chartarea}
  width={Browser.isDevice ? '100%' : '80%'}
  title='AAPL 2012-2017'>
  <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, StochasticIndicator]} />
  <AxesDirective>
    <AxisDirective rowIndex={0} name='secondary' opposedPosition={true}
      majorGridLines={lines} majorTickLines={lines}
      minimum={0} maximum={120} interval={60} title='Stochastic'
lineStyle={lines}>
    </AxisDirective>
  </AxesDirective>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={chartData} width={2}
      xName='x' yName='y' low='low' high='high' close='close'
volume='volume' open='open'
      name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
type='Candle' animation={animation}>
    </SeriesDirective>
  </SeriesCollectionDirective>
  <IndicatorsDirective>
    <IndicatorDirective type='Stochastic' field='Close'
seriesName='Apple Inc' yAxisName='secondary' fill='#6063ff' kPeriod={2}
dPeriod={3} showZones={true} periodLine={periodline} period={3}
animation={animation1} upperLine={upperline} lowerLine={lowerline}>
    </IndicatorDirective>
  </IndicatorsDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Customization of StochasticIndicator

stroke, stroke-width, and color of upperLine can be customized by using [upperLine](#), the lowerLine can be customized by using [lowerLine](#) and the periodLine can be customized by using [periodLine](#) properties of indicator. To customize the period to find the average price using [kPeriod](#) and [dPeriod](#) properties.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';

```



```

import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, RowDirective, RowsDirective, SeriesDirective, Inject,
StripLine, DateTime, Logarithmic, Tooltip, CandleSeries, Crosshair,
StripLinesDirective, StripLineDirective, LineSeries, StochasticIndicator,
IndicatorsDirective, IndicatorDirective, Category } from '@syncfusion/ej2-
react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'DateTime', intervalType: 'Months',
majorGridLines: { width: 0 }, crosshairTooltip: { enable: true } };
    const primaryyAxis = { title: 'Price', labelFormat: '${value}', minimum:
30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
    const animation = { enable: true };
    const chartarea = { border: { width: 0 } };
    const crosshair = { enable: true, lineType: 'Vertical' };
    const tooltip = { enable: true, shared: true };
    const lines = { width: 0 };
    const animation1 = { enable: false };
    const upperline = { color: '#e74c3d' };
    const lowerline = { color: '#2ecd71' };
    const periodline = { color: '#f2ec2f' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} crosshair={crosshair}
chartArea={chartarea} width={Browser.isDevice ? '100%' : '80%'} title='AAPL
2012-2017'>
        <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, StochasticIndicator]}/>
        <RowsDirective>
            <RowDirective height={'40%'}></RowDirective>
            <RowDirective height={'60%'}></RowDirective>
        </RowsDirective>
        <AxesDirective>
            <AxisDirective rowIndex={0} name='secondary' opposedPosition={true}
majorGridLines={lines} majorTickLines={lines} minimum={0} maximum={120}
interval={60} title='Stochastic' lineStyle={lines}>
                <StripLinesDirective>
                    <StripLineDirective start={0} end={120} text='' color='black'
visible={true} opacity={0.03} zIndex='Behind'>
                    </StripLineDirective>
                </StripLinesDirective>
            </AxisDirective>
        </AxesDirective>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} width={2} xName='x'
yName='y' low='low' high='high' close='close' volume='volume' open='open'
name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
type='Candle' animation={animation}>
            </SeriesDirective>
        </SeriesCollectionDirective>
        <IndicatorsDirective>
            <IndicatorDirective type='Stochastic' field='Close'
seriesName='Apple Inc' yAxisName='secondary' fill='#6063ff' kPeriod={2}
dPeriod={3} showZones={true} periodLine={periodline} period={3}
animation={animation1} upperLine={upperline} lowerLine={lowerline}>
            </IndicatorDirective>
        </IndicatorsDirective>
    </ChartComponent>;

```

```

}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
    ChartComponent, SeriesCollectionDirective, AxesDirective, AxisDirective,
    RowDirective, RowsDirective, SeriesDirective, Inject,
    StripLine, DateTime, Logarithmic, Tooltip, CandleSeries, Crosshair,
    Zoom, ILoadedEventArgs, StripLinesDirective, StripLineDirective,
    LineSeries, StochasticIndicator, IndicatorsDirective,
    IndicatorDirective, Category, ChartTheme,
    AxisModel, ChartAreaModel, CrosshairSettingsModel, TooltipSettingsModel
} from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime', intervalType:
'Months', majorGridLines: { width: 0 }, crosshairTooltip: { enable: true }
};
    const primaryyAxis: AxisModel = { title: 'Price', labelFormat: '${value}',
minimum: 30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
    const animation = { enable: true };
    const chartarea: ChartAreaModel = { border: { width: 0 } };
    const crosshair: CrosshairSettingsModel = { enable: true, lineType:
'Vertical' };
    const tooltip: TooltipSettingsModel = { enable: true, shared: true };
    const lines = { width: 0 };
    const animation1 = { enable: false };
    const upperline = { color: '#e74c3d' };
    const lowerline = { color: '#2ecd71' };
    const periodline = { color: '#f2ec2f' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        tooltip={tooltip}
        crosshair={crosshair}
        chartArea={chartarea}
        width={Browser.isDevice ? '100%' : '80%'}
        title='AAPL 2012-2017'>
        <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries, StochasticIndicator]} />
        <RowsDirective>
            <RowDirective height={'40%'}></RowDirective>
            <RowDirective height={'60%'}></RowDirective>
        </RowsDirective>
        <AxesDirective>
            <AxisDirective rowIndex={0} name='secondary' opposedPosition={true}
                majorGridLines={lines} majorTickLines={lines}
                minimum={0} maximum={120} interval={60} title='Stochastic'
                lineStyle={lines}>
                <StripLinesDirective>

```

```

        <StripLineDirective start={0} end={120} text='' color='black'
visible={true} opacity={0.03} zIndex='Behind'>
        </StripLineDirective>
    </StripLinesDirective>
</AxisDirective>
</AxesDirective>
<SeriesCollectionDirective>
    <SeriesDirective dataSource={chartData} width={2}
        xName='x' yName='y' low='low' high='high' close='close'
volume='volume' open='open'
        name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
type='Candle' animation={animation}>
    </SeriesDirective>
</SeriesCollectionDirective>
<IndicatorsDirective>
    <IndicatorDirective type='Stochastic' field='Close'
seriesName='Apple Inc' yAxisName='secondary' fill='#6063ff' kPeriod={2}
dPeriod={3} showZones={true} periodLine={periodline} period={3}
animation={animation1} upperLine={upperline} lowerLine={lowerline}>
    </IndicatorDirective>
</IndicatorsDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Triangular Moving Average (TMA)

Moving average Indicators are used to define the direction of the trend. To render a TMA Indicator, use indicator [type](#) as `Tma` and inject `TmaIndicator` module into services.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
CandleSeries, Category, Tooltip, DateTime, Logarithmic, Crosshair,
LineSeries, TmaIndicator, IndicatorsDirective, IndicatorDirective } from
'@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'DateTime', intervalType: 'Months',
majorGridLines: { width: 0 }, crosshairTooltip: { enable: true } };
    const primaryyAxis = { title: 'Price', labelFormat: '${value}', minimum:
30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
    const animation = { enable: true };
    const chartarea = { border: { width: 0 } };
    const crosshair = { enable: true, lineType: 'Vertical' };
    const tooltip = { enable: true, shared: true };
    const animation1 = { enable: false };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} crosshair={crosshair}
chartArea={chartarea} width={Browser.isDevice ? '100%' : '80%'} title='AAPL
2012-2017'>
        <Inject services={[CandleSeries, Category, Tooltip, DateTime,
Logarithmic, Crosshair, LineSeries, TmaIndicator]}/>
    </ChartComponent>

```

```

    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} width={2} xName='x'
yName='y' low='low' high='high' close='close' volume='volume' open='open'
name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
type='Candle' animation={animation1}>
    </SeriesDirective>
  </SeriesCollectionDirective>
  <IndicatorsDirective>
    <IndicatorDirective type='Tma' field='Close' seriesName='Apple Inc'
fill='#6063ff' period={14} animation={animation}>
    </IndicatorDirective>
  </IndicatorsDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
  ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
  CandleSeries, Category, Tooltip, DateTime,
  Zoom, Logarithmic, Crosshair, LineSeries, TmaIndicator,
  IndicatorsDirective, IndicatorDirective, ILoadedEventArgs,
  ChartTheme, AxisModel, ChartAreaModel, CrosshairSettingsModel,
  TooltipSettingsModel
} from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime', intervalType:
'Months', majorGridLines: { width: 0 }, crosshairTooltip: { enable: true }
};
  const primaryyAxis: AxisModel = { title: 'Price', labelFormat: '${value}',
minimum: 30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
  const animation = { enable: true };
  const chartarea: ChartAreaModel = { border: { width: 0 } };
  const crosshair: CrosshairSettingsModel = { enable: true, lineType:
'Vertical' };
  const tooltip: TooltipSettingsModel = { enable: true, shared: true };
  const animation1 = { enable: false };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    tooltip={tooltip}
    crosshair={crosshair}
    chartArea={chartarea}
    width={Browser.isDevice ? '100%' : '80%'}
    title='AAPL 2012-2017'>
    <Inject services={[CandleSeries, Category, Tooltip, DateTime,
Logarithmic, Crosshair, LineSeries, TmaIndicator]} />
    <SeriesCollectionDirective>

```

```

    <SeriesDirective dataSource={chartData} width={2} xName='x'
    yName='y' low='low' high='high' close='close' volume='volume' open='open'
    name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
    type='Candle' animation={animation1}>
    </SeriesDirective>
  </SeriesCollectionDirective>
  <IndicatorsDirective>
    <IndicatorDirective type='Tma' field='Close' seriesName='Apple Inc'
    fill='#6063ff' period={14} animation={animation}>
    </IndicatorDirective>
  </IndicatorsDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Customization of Technical Indicators

stroke, stroke-width, and color of signalLine can be customized by using [fill](#), [width](#) and [dashArray](#) properties. and the [period](#) property is used to predict the data forecast calculations. The [field](#) value is used to compare the current price with previous price. It is applicable to Bollinger bands and moving averages. The [showZones](#) property is used to show/hides the overBought and overSold regions. It is applicable for RSI and stochastic indicators.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
CandleSeries, Category, Tooltip, DateTime, Logarithmic, Crosshair,
LineSeries, TmaIndicator, IndicatorsDirective, IndicatorDirective } from
'@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
  const primaryXAxis = { valueType: 'DateTime', intervalType: 'Months',
majorGridLines: { width: 0 }, crosshairTooltip: { enable: true } };
  const primaryYAxis = { title: 'Price', labelFormat: '${value}', minimum:
30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
  const animation = { enable: true };
  const chartarea = { border: { width: 0 } };
  const crosshair = { enable: true, lineType: 'Vertical' };
  const tooltip = { enable: true, shared: true };
  const animation1 = { enable: false };
  return <ChartComponent id='charts' primaryXAxis={primaryXAxis}
primaryYAxis={primaryYAxis} tooltip={tooltip} crosshair={crosshair}
chartArea={chartarea} width={Browser.isDevice ? '100%' : '80%'} title='AAPL
2012-2017'>
    <Inject services={[CandleSeries, Category, Tooltip, DateTime,
Logarithmic, Crosshair, LineSeries, TmaIndicator]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} width={2} xName='x'
      yName='y' low='low' high='high' close='close' volume='volume' open='open'
      name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
      type='Candle' animation={animation1}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>

```

```

        </SeriesCollectionDirective>
        <IndicatorsDirective>
            <IndicatorDirective type='Tma' field='Close' seriesName='Apple Inc'
fill='red' period={3} animation={animation}>
            </IndicatorDirective>
        </IndicatorsDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
    ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
    CandleSeries, Category, Tooltip, DateTime,
    Zoom, Logarithmic, Crosshair, LineSeries, TmaIndicator,
    IndicatorsDirective, IndicatorDirective, ILoadedEventArgs,
    ChartTheme, AxisModel, ChartAreaModel, CrosshairSettingsModel,
    TooltipSettingsModel
} from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime', intervalType:
'Months', majorGridLines: { width: 0 }, crosshairTooltip: { enable: true }
};
    const primaryyAxis: AxisModel = { title: 'Price', labelFormat: '${value}',
minimum: 30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
    const animation = { enable: true };
    const chartarea: ChartAreaModel = { border: { width: 0 } };
    const crosshair: CrosshairSettingsModel = { enable: true, lineType:
'Vertical' };
    const tooltip: TooltipSettingsModel = { enable: true, shared: true };
    const animation1 = { enable: false };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        tooltip={tooltip}
        crosshair={crosshair}
        chartArea={chartarea}
        width={Browser.isDevice ? '100%' : '80%'}
        title='AAPL 2012-2017'>
        <Inject services={[CandleSeries, Category, Tooltip, DateTime,
Logarithmic, Crosshair, LineSeries, TmaIndicator]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} width={2} xName='x'
yName='y' low='low' high='high' close='close' volume='volume' open='open'
name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
type='Candle' animation={animation1}>
            </SeriesDirective>
        </SeriesCollectionDirective>
        <IndicatorsDirective>

```

```

        <IndicatorDirective type='Tma' field='Close' seriesName='Apple Inc'
        fill='red' period={3} animation={animation}>
        </IndicatorDirective>
    </IndicatorsDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Data Source

Usually technical indicators are added along with a financial series. The [seriesName](#) represents the series, the data of which has to be analysed through indicators.

Technical indicators can also be added without series using [dataSource](#) property of indicator.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, CandleSeries, Category, Tooltip,
DateTime, Logarithmic, Crosshair, LineSeries,
AccumulationDistributionIndicator, StripLine, IndicatorsDirective,
IndicatorDirective } from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const axisLabelRender = (args) => {
        if (args.axis.name === 'secondary') {
            let value = Number(args.text) / 1000000000;
            args.text = String(value) + 'bn';
        }
    };
    const primaryxAxis = { valueType: 'DateTime', intervalType: 'Months',
majorGridLines: { width: 0 }, crosshairTooltip: { enable: true } };
    const primaryyAxis = { title: 'Price', labelFormat: '${value}', minimum:
30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
    const animation = { enable: true };
    const chartarea = { border: { width: 0 } };
    const crosshair = { enable: true, lineType: 'Vertical' };
    const tooltip = { enable: true, shared: true };
    const lines = { width: 0 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} crosshair={crosshair}
axisLabelRender={axisLabelRender.bind(this)} chartArea={chartarea}
width={Browser.isDevice ? '100%' : '80%'} title='AAPL 2012-2017'>
        <Inject services={[CandleSeries, Category, Tooltip, StripLine,
DateTime, Logarithmic, Crosshair, LineSeries,
AccumulationDistributionIndicator]} />
        <AxesDirective>
            <AxisDirective rowIndex={0} name='secondary' opposedPosition={true}
majorGridLines={lines} majorTickLines={lines} minimum={-7000000000}
maximum={5000000000} interval={6000000000} title='Accumulation Distribution'
lineStyle={lines}>
            </AxisDirective>
        </AxesDirective>

```

```

        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} width={2} xName='x'
yName='y' low='low' high='high' close='close' volume='volume' open='open'
name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
type='Candle' animation={animation}>
            </SeriesDirective>
        </SeriesCollectionDirective>
        <IndicatorsDirective>
            <IndicatorDirective type='AccumulationDistribution' field='Close'
seriesName='Apple Inc' yAxisName='secondary' fill='#6063ff' period={3}
animation={animation}>
            </IndicatorDirective>
        </IndicatorsDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, RowDirective, RowsDirective, SeriesDirective, Inject,
CandleSeries, Category, Tooltip, ILoadedEventArgs, DateTime, Zoom,
Logarithmic, StripLineDirective,
Crosshair, LineSeries, AccumulationDistributionIndicator,
IAxisLabelRenderEventArgs,
StripLine, ChartTheme, IndicatorsDirective, IndicatorDirective,
StripLinesDirective, AxisModel, ChartAreaModel, CrosshairSettingsModel,
TooltipSettingsModel }
from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const axisLabelRender = (args: IAxisLabelRenderEventArgs): void => {
        if (args.axis.name === 'secondary') {
            let value: number = Number(args.text) / 1000000000;
            args.text = String(value) + 'bn';
        }
    }
    const primaryXAxis: AxisModel = { valueType: 'DateTime', intervalType:
'Months', majorGridLines: { width: 0 }, crosshairTooltip: { enable: true }
};
    const primaryYAxis: AxisModel = { title: 'Price', labelFormat: '${value}',
minimum: 30, maximum: 180, interval: 30, lineStyle: { width: 0 } };
    const animation = { enable: true };
    const chartarea: ChartAreaModel = { border: { width: 0 } };
    const crosshair: CrosshairSettingsModel = { enable: true, lineType:
'Vertical' };
    const tooltip: TooltipSettingsModel = { enable: true, shared: true };
    const lines = { width: 0 };
    return <ChartComponent id='charts'
        primaryXAxis={primaryXAxis}
        primaryYAxis={primaryYAxis}

```



```

    tooltip={tooltip}
    crosshair={crosshair}
    axisLabelRender={axisLabelRender.bind(this)}
    chartArea={chartArea}
    width={Browser.isDevice ? '100%' : '80%'}
    title='AAPL 2012-2017'>
    <Inject services={[CandleSeries, Category, Tooltip, StripLine,
    DateTime, Logarithmic, Crosshair, LineSeries,
    AccumulationDistributionIndicator]} />
    <AxesDirective>
      <AxisDirective rowIndex={0} name='secondary' opposedPosition={true}
        majorGridLines={lines} majorTickLines={lines} minimum={-
7000000000} maximum={5000000000}
        interval={6000000000} title='Accumulation Distribution'
lineStyle={lines}>
      </AxisDirective>
    </AxesDirective>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={chartData} width={2}
        xName='x' yName='y' low='low' high='high' close='close'
volume='volume' open='open'
        name='Apple Inc' bearFillColor='#2ecd71' bullFillColor='#e74c3d'
type='Candle' animation={animation}>
      </SeriesDirective>
    </SeriesCollectionDirective>
    <IndicatorsDirective>
      <IndicatorDirective type='AccumulationDistribution' field='Close'
seriesName='Apple Inc' yAxisName='secondary' fill='#6063ff' period={3}
animation={animation}>
      </IndicatorDirective>
    </IndicatorsDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Module Dependencies

To render a Indicator, it is mandatory to inject `LineSeries` module using `Chart.Inject(LineSeries)`.

In addition to that, MACD Indicator requires `ColumnSeries` and BollingerBands requires `RangeAreaSeries`.

<!-- markdownlint-disable MD036 -->

Trend lines in React Chart component

Trendlines are used to show the direction and speed of price.

Trendlines can be generated for Cartesian type series (Line, Column, Scatter, Area, Candle, Hilo etc.) except bar type series. You can add more than one trendline to a series.

Chart supports 6 types of trendlines.

Linear

A linear trendline is a best fit straight line that is used with simpler data sets. To render a linear trendline, use trendline `type` as `Linear` and inject `Trendlines` module using `Chart.Inject(Trendlines)`.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective,
TrendlineDirective, TrendlinesDirective, Inject, Tooltip, LineSeries,
ScatterSeries, SplineSeries, Trendlines, Category } from '@syncfusion/ej2-
react-charts';
function App() {
    const series1 = [];
    const yValue = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89, 8.68,
9.48, 10.11, 11.36, 12.34, 12.60, 12.95,
13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52,
36.36,
41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17,
41.20, 43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
    function chartLoad() {
        let i;
        let j = 0;
        for (i = 1973; i <= 2013; i++) {
            series1.push({ x: i, y: yValue[j] });
            j++;
        }
    }
    const primaryxAxis = { title: 'Months', majorGridLines: { width: 0 } };
    const primaryyAxis = {
        title: 'Rupees against Dollars', interval: 10,
        lineStyle: { width: 0 }, majorTickLines: { width: 0 }
    };
    const tooltip = { enable: true };
    const chartarea = { border: { width: 0 } };
    const marker = { visible: false };
    chartLoad();
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} chartArea={chartarea}
title='Historical Indian Rupee Rate (INR USD)'>
        <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={series1} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
                <TrendlinesDirective>
                    <TrendlineDirective type='Linear' width={3} marker={marker}>
                    </TrendlineDirective>
                </TrendlinesDirective>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";

```

```

import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
    ChartComponent, SeriesCollectionDirective, SeriesDirective,
    TrendlineDirective, TrendlinesDirective, Inject,
    Tooltip, LineSeries, ScatterSeries, SplineSeries, Trendlines, Category,
    ChartTheme
    ,AxisModel,TooltipSettingsModel,ChartAreaModel
} from '@syncfusion/ej2-react-charts';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import { NumericTextBoxComponent } from '@syncfusion/ej2-react-inputs';
import { SampleBase } from '../common/sample-base';
import { PropertyPane } from '../common/property-pane';
import { EmitType } from '@syncfusion/ej2-base';
function App() {
    const series1: Object[] = [];
    const yValue: number[] = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89,
    8.68, 9.48, 10.11, 11.36, 12.34, 12.60, 12.95,
    13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52, 36.36,
    41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17, 41.20,
    43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
    function chartLoad(): void {
        let i: number;
        let j: number = 0;
        for (i = 1973; i <= 2013; i++) {
            series1.push({ x: i, y: yValue[j] });
            j++;
        }
    }
    const primaryxAxis: AxisModel = { title: 'Months', majorGridLines: {
width: 0 } };
    const primaryyAxis: AxisModel = {
        title: 'Rupees against Dollars', interval: 10,
        lineStyle: { width: 0 }, majorTickLines: { width: 0 }
    };
    const tooltip: TooltipSettingsModel = { enable: true };
    const chartarea: ChartAreaModel = { border: { width: 0 } };
    const marker = { visible: false };
    chartLoad();
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        tooltip={tooltip}
        chartArea={chartarea}
        title='Historical Indian Rupee Rate (INR USD)'>
        <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={series1} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
                <TrendlinesDirective>
                    <TrendlineDirective type='Linear' width={3} marker={marker}>
                    </TrendlineDirective>
                </TrendlinesDirective>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>

```

```
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Exponential

An exponential trendline is a curved line that is most useful when data values rise or fall at increasingly higher rates. You cannot create an exponential trendline, if your data contains zero or negative values.

To render a exponential trendline, use trendline [type](#) as **Exponential** and inject **Trendlines** module using **Chart.Inject(Trendlines)**.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective,
TrendlineDirective, TrendlinesDirective, Inject, Tooltip, LineSeries,
ScatterSeries, SplineSeries, Trendlines, Category } from '@syncfusion/ej2-
react-charts';
function App() {
    const series1 = [];
    const yValue = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89, 8.68,
9.48, 10.11, 11.36, 12.34, 12.60, 12.95,
13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52,
36.36,
41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17,
41.20, 43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
    function chartLoad() {
        let i;
        let j = 0;
        for (i = 1973; i <= 2013; i++) {
            series1.push({ x: i, y: yValue[j] });
            j++;
        }
    }
    const primaryxAxis = { title: 'Months', majorGridLines: { width: 0 } };
    const primaryyAxis = {
        title: 'Rupees against Dollars', interval: 10,
        lineStyle: { width: 0 }, majorTickLines: { width: 0 }
    };
    const tooltip = { enable: true };
    const chartarea = { border: { width: 0 } };
    const marker = { visible: false };
    chartLoad();
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} chartArea={chartarea}
title='Historical Indian Rupee Rate (INR USD)'>
        <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={series1} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
                <TrendlinesDirective>
                    <TrendlineDirective type='Exponential' width={3}
marker={marker}>
                        </TrendlineDirective>
```

```

        </TrendlinesDirective>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
  ChartComponent, SeriesCollectionDirective, SeriesDirective,
  TrendlineDirective, TrendlinesDirective, Inject,
  Tooltip, LineSeries, ScatterSeries, SplineSeries, Trendlines, Category,
  ChartTheme
  ,AxisModel,TooltipSettingsModel,ChartAreaModel
} from '@syncfusion/ej2-react-charts';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import { NumericTextBoxComponent } from '@syncfusion/ej2-react-inputs';
import { SampleBase } from '../common/sample-base';
import { PropertyPane } from '../common/property-pane';
import { EmitType } from '@syncfusion/ej2-base';
function App() {
  const series1: Object[] = [];
  const yValue: number[] = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89,
    8.68, 9.48, 10.11, 11.36, 12.34, 12.60, 12.95,
    13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52, 36.36,
    41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17, 41.20,
    43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
  function chartLoad(): void {
    let i: number;
    let j: number = 0;
    for (i = 1973; i <= 2013; i++) {
      series1.push({ x: i, y: yValue[j] });
      j++;
    }
  }
  const primaryxAxis: AxisModel = { title: 'Months', majorGridLines: {
width: 0 } };
  const primaryyAxis: AxisModel = {
    title: 'Rupees against Dollars', interval: 10,
    lineStyle: { width: 0 }, majorTickLines: { width: 0 }
  };
  const tooltip: TooltipSettingsModel = { enable: true };
  const chartarea: ChartAreaModel = { border: { width: 0 } };
  const marker = { visible: false };
  chartLoad();
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    tooltip={tooltip}
    chartArea={chartarea}

```

```

        title='Historical Indian Rupee Rate (INR USD) '>
        <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={series1} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
                <TrendlinesDirective>
                    <TrendlineDirective type='Exponential' width={3}
marker={marker}>>
                </TrendlineDirective>
            </TrendlinesDirective>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Logarithmic

A logarithmic trendline is a best-fit curved line that is most useful when the rate of change in the data increases or decreases quickly and then levels out. A logarithmic trendline can use negative and/or positive values.

To render a logarithmic trendline, use trendline [type](#) as `Logarithmic` and inject `Trendlines` module using `Chart.Inject(Trendlines)`.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective,
TrendlineDirective, TrendlinesDirective, Inject, Tooltip, LineSeries,
ScatterSeries, SplineSeries, Trendlines, Category } from '@syncfusion/ej2-
react-charts';
function App() {
    const series1 = [];
    const yValue = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89, 8.68,
9.48, 10.11, 11.36, 12.34, 12.60, 12.95,
13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52,
36.36,
41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17,
41.20, 43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
    function chartLoad() {
        let i;
        let j = 0;
        for (i = 1973; i <= 2013; i++) {
            series1.push({ x: i, y: yValue[j] });
            j++;
        }
    }
    const primaryxAxis = { title: 'Months', majorGridLines: { width: 0 } };
    const primaryyAxis = {
        title: 'Rupees against Dollars', interval: 10,
        lineStyle: { width: 0 }, majorTickLines: { width: 0 }
    };
};

```

```

const tooltip = { enable: true };
const chartarea = { border: { width: 0 } };
const marker = { visible: false };
chartLoad();
return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} chartArea={chartarea}
title='Historical Indian Rupee Rate (INR USD)'>
  <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]}/>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={series1} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
      <TrendlinesDirective>
        <TrendlineDirective type='Logarithmic' width={3}
marker={marker}>
          </TrendlineDirective>
        </TrendlinesDirective>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
  ChartComponent, SeriesCollectionDirective, SeriesDirective,
  TrendlineDirective, TrendlinesDirective, Inject,
  Tooltip, LineSeries, ScatterSeries, SplineSeries, Trendlines, Category,
  ChartTheme
, AxisModel, TooltipSettingsModel, ChartAreaModel
} from '@syncfusion/ej2-react-charts';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import { NumericTextBoxComponent } from '@syncfusion/ej2-react-inputs';
import { SampleBase } from '../common/sample-base';
import { PropertyPane } from '../common/property-pane';
import { EmitType } from '@syncfusion/ej2-base';
function App() {
  const series1: Object[] = [];
  const yValue: number[] = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89,
8.68, 9.48, 10.11, 11.36, 12.34, 12.60, 12.95,
13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52, 36.36,
41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17, 41.20,
43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
  function chartLoad(): void {
    let i: number;
    let j: number = 0;
    for (i = 1973; i <= 2013; i++) {
      series1.push({ x: i, y: yValue[j] });
      j++;
    }
  }
}

```

```

    }
    const primaryxAxis: AxisModel = { title: 'Months', majorGridLines: {
width: 0 } };
    const primaryyAxis: AxisModel = {
      title: 'Rupees against Dollars', interval: 10,
      lineStyle: { width: 0 }, majorTickLines: { width: 0 }
    };
    const tooltip: TooltipSettingsModel = { enable: true };
    const chartarea: ChartAreaModel = { border: { width: 0 } };
    const marker = { visible: false };
    chartLoad();
    return <ChartComponent id='charts'
      primaryXAxis={primaryxAxis}
      primaryYAxis={primaryyAxis}
      tooltip={tooltip}
      chartArea={chartarea}
      title='Historical Indian Rupee Rate (INR USD)'>
      <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]} />
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={series1} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
          <TrendlinesDirective>
            <TrendlineDirective type='Logarithmic' width={3}
marker={marker}>
            </TrendlineDirective>
          </TrendlinesDirective>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  </>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Polynomial

A polynomial trendline is a curved line that is used when data fluctuates.

To render a polynomial trendline, use trendline [type](#) as **Polynomial** and inject **Trendlines** module using **Chart.Inject(Trendlines)**.

polynomialOrder used to define the polynomial value.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective,
TrendlineDirective, TrendlinesDirective, Inject, Tooltip, LineSeries,
ScatterSeries, SplineSeries, Trendlines, Category } from '@syncfusion/ej2-
react-charts';
function App() {
  const series1 = [];
  const yValue = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89, 8.68,
9.48, 10.11, 11.36, 12.34, 12.60, 12.95,

```



```

13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52,
36.36,
41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17,
41.20, 43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
function chartLoad() {
    let i;
    let j = 0;
    for (i = 1973; i <= 2013; i++) {
        series1.push({ x: i, y: yValue[j] });
        j++;
    }
}
const primaryxAxis = { title: 'Months', majorGridLines: { width: 0 } };
const primaryyAxis = {
    title: 'Rupees against Dollars', interval: 10,
    lineStyle: { width: 0 }, majorTickLines: { width: 0 }
};
const tooltip = { enable: true };
const chartarea = { border: { width: 0 } };
const marker = { visible: false };
chartLoad();
return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} chartArea={chartarea}
title='Historical Indian Rupee Rate (INR USD)'>
    <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]}/>
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={series1} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
            <TrendlinesDirective>
                <TrendlineDirective type='Polynomial' width={3} marker={marker}>
                </TrendlineDirective>
            </TrendlinesDirective>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
    ChartComponent, SeriesCollectionDirective, SeriesDirective,
    TrendlineDirective, TrendlinesDirective, Inject,
    Tooltip, LineSeries, ScatterSeries, SplineSeries, Trendlines, Category,
    ChartTheme
    ,AxisModel,TooltipSettingsModel,ChartAreaModel
} from '@syncfusion/ej2-react-charts';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import { NumericTextBoxComponent } from '@syncfusion/ej2-react-inputs';
import { SampleBase } from '../common/sample-base';

```

```

import { PropertyPane } from '../common/property-pane';
import { EmitType } from '@syncfusion/ej2-base';
function App() {
    const series1: Object[] = [];
    const yValue: number[] = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89,
8.68, 9.48, 10.11, 11.36, 12.34, 12.60, 12.95,
    13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52, 36.36,
    41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17, 41.20,
43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
    function chartLoad(): void {
        let i: number;
        let j: number = 0;
        for (i = 1973; i <= 2013; i++) {
            series1.push({ x: i, y: yValue[j] });
            j++;
        }
    }
    const primaryxAxis: AxisModel = { title: 'Months', majorGridLines: {
width: 0 } };
    const primaryyAxis: AxisModel = {
        title: 'Rupees against Dollars', interval: 10,
        lineStyle: { width: 0 }, majorTickLines: { width: 0 }
    };
    const tooltip: TooltipSettingsModel = { enable: true };
    const chartarea: ChartAreaModel = { border: { width: 0 } };
    const marker = { visible: false };
    chartLoad();
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        tooltip={tooltip}
        chartArea={chartarea}
        title='Historical Indian Rupee Rate (INR USD)'>
        <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={series1} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
                <TrendlinesDirective>
                    <TrendlineDirective type='Polynomial' width={3} marker={marker}>
                    </TrendlineDirective>
                </TrendlinesDirective>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
    </return>
    </function>
    </App>
    </export>
    </ReactDOM.render>
    </document.getElementById>
    </charts>
    </>

```

Power

A power trendline is a curved line that is best used with data sets that compare measurements that increase at a specific rate.

To render a power trendline, use trendline [type](#) as **Power** and inject **Trendlines** module using `Chart.Inject(Trendlines)`.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective,
TrendlineDirective, TrendlinesDirective, Inject, Tooltip, LineSeries,
ScatterSeries, SplineSeries, Trendlines, Category } from '@syncfusion/ej2-
react-charts';
function App() {
    const powerData = [
        { x: 1, y: 10 }, { x: 2, y: 50 }, { x: 3, y: 80 }, { x: 4, y: 110 },
        { x: 5, y: 180 }, { x: 6, y: 220 }, { x: 7, y: 300 }, { x: 8, y: 370
    }, { x: 9, y: 490 }, { x: 10, y: 500 }
    ];
    const primaryxAxis = { title: 'Months', majorGridLines: { width: 0 } };
    const primaryyAxis = {
        title: 'Rupees against Dollars', interval: 50,
        lineStyle: { width: 0 }, majorTickLines: { width: 0 }
    };
    const tooltip = { enable: true };
    const chartarea = { border: { width: 0 } };
    const marker = { visible: false };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} chartArea={chartarea}
title='Historical Indian Rupee Rate (INR USD)'>
        <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={powerData} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
                <TrendlinesDirective>
                    <TrendlineDirective type='Power' width={3} marker={marker}>
                    </TrendlineDirective>
                </TrendlinesDirective>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
    ChartComponent, SeriesCollectionDirective, SeriesDirective,
    TrendlineDirective, TrendlinesDirective, Inject,
    Tooltip, LineSeries, ScatterSeries, SplineSeries, Trendlines, Category,
    ChartTheme
    ,AxisModel,TooltipSettingsModel,ChartAreaModel
```

```

} from '@syncfusion/ej2-react-charts';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import { NumericTextBoxComponent } from '@syncfusion/ej2-react-inputs';
import { SampleBase } from '../common/sample-base';
import { PropertyPane } from '../common/property-pane';
import { EmitType } from '@syncfusion/ej2-base';
function App() {
  const powerData: object[] = [
    { x: 1, y: 10 }, { x: 2, y: 50 }, { x: 3, y: 80 }, { x: 4, y: 110 },
    { x: 5, y: 180 }, { x: 6, y: 220 }, { x: 7, y: 300 }, { x: 8, y: 370 },
    { x: 9, y: 490 }, { x: 10, y: 500 }
  ];
  const primaryxAxis: AxisModel = { title: 'Months', majorGridLines: {
width: 0 } };
  const primaryyAxis: AxisModel = {
    title: 'Rupees against Dollars', interval: 50,
    lineStyle: { width: 0 }, majorTickLines: { width: 0 }
  };
  const tooltip: TooltipSettingsModel = { enable: true };
  const chartarea: ChartAreaModel = { border: { width: 0 } };
  const marker = { visible: false };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    tooltip={tooltip}
    chartArea={chartarea}
    title='Historical Indian Rupee Rate (INR USD)'>
    <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={powerData} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
        <TrendlinesDirective>
          <TrendlineDirective type='Power' width={3} marker={marker}>
            </TrendlineDirective>
          </TrendlinesDirective>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  </>;
}
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Moving Average

A moving average trendline smoothen out fluctuations in data to show a pattern or trend more clearly.

To render a moving average trendline, use trendline [type](#) as `MovingAverage` and inject `Trendlines` module using `Chart.Inject(Trendlines)`.

`period` property defines the period to find the moving average.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";

```

```

import { ChartComponent, SeriesCollectionDirective, SeriesDirective,
TrendlineDirective, TrendlinesDirective, Inject, Tooltip, LineSeries,
ScatterSeries, SplineSeries, Trendlines, Category } from '@syncfusion/ej2-
react-charts';
function App() {
  const series1 = [];
  const yValue = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89, 8.68,
9.48, 10.11, 11.36, 12.34, 12.60, 12.95,
13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52,
36.36,
41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17,
41.20, 43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
  function chartLoad() {
    let i;
    let j = 0;
    for (i = 1973; i <= 2013; i++) {
      series1.push({ x: i, y: yValue[j] });
      j++;
    }
  }
  const primaryxAxis = { title: 'Months', majorGridLines: { width: 0 } };
  const primaryyAxis = {
    title: 'Rupees against Dollars', interval: 10,
    lineStyle: { width: 0 }, majorTickLines: { width: 0 }
  };
  const tooltip = { enable: true };
  const chartarea = { border: { width: 0 } };
  const marker = { visible: false };
  chartLoad();
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} chartArea={chartarea}
title='Historical Indian Rupee Rate (INR USD)'>
    <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={series1} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
        <TrendlinesDirective>
          <TrendlineDirective type='MovingAverage' width={3}
marker={marker}>
          </TrendlineDirective>
        </TrendlinesDirective>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {

```

```

    ChartComponent, SeriesCollectionDirective, SeriesDirective,
    TrendlineDirective, TrendlinesDirective, Inject,
    Tooltip, LineSeries, ScatterSeries, SplineSeries, Trendlines, Category,
    ChartTheme
    ,AxisModel,TooltipSettingsModel,ChartAreaModel
} from '@syncfusion/ej2-react-charts';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import { NumericTextBoxComponent } from '@syncfusion/ej2-react-inputs';
import { SampleBase } from '../common/sample-base';
import { PropertyPane } from '../common/property-pane';
import { EmitType } from '@syncfusion/ej2-base';
function App() {
    const series1: Object[] = [];
    const yValue: number[] = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89,
8.68, 9.48, 10.11, 11.36, 12.34, 12.60, 12.95,
    13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52, 36.36,
    41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17, 41.20,
43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
    function chartLoad(): void {
        let i: number;
        let j: number = 0;
        for (i = 1973; i <= 2013; i++) {
            series1.push({ x: i, y: yValue[j] });
            j++;
        }
    }
    const primaryxAxis: AxisModel = { title: 'Months', majorGridLines: {
width: 0 } };
    const primaryyAxis: AxisModel = {
        title: 'Rupees against Dollars', interval: 10,
        lineStyle: { width: 0 }, majorTickLines: { width: 0 }
    };
    const tooltip: TooltipSettingsModel = { enable: true };
    const chartarea: ChartAreaModel = { border: { width: 0 } };
    const marker = { visible: false };
    chartLoad();
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        tooltip={tooltip}
        chartArea={chartarea}
        title='Historical Indian Rupee Rate (INR USD)'>
        <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={series1} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
                <TrendlinesDirective>
                    <TrendlineDirective type='MovingAverage' width={3}
marker={marker}>
                        </TrendlineDirective>
                    </TrendlinesDirective>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>
    </return>
};
export default App;

```

```
ReactDOM.render(<App />, document.getElementById("charts"));
```

Customization of Trendline

The [fill](#) and [width](#) properties are used to customize the appearance of the trendline.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective,
TrendlineDirective, TrendlinesDirective, Inject, Tooltip, LineSeries,
ScatterSeries, SplineSeries, Trendlines, Category } from '@syncfusion/ej2-
react-charts';
function App() {
    const series1 = [];
    const yValue = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89, 8.68,
9.48, 10.11, 11.36, 12.34, 12.60, 12.95,
13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52,
36.36,
41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17,
41.20, 43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
    function chartLoad() {
        let i;
        let j = 0;
        for (i = 1973; i <= 2013; i++) {
            series1.push({ x: i, y: yValue[j] });
            j++;
        }
    }
    const primaryxAxis = { title: 'Months', majorGridLines: { width: 0 } };
    const primaryyAxis = {
        title: 'Rupees against Dollars', interval: 10,
        lineStyle: { width: 0 }, majorTickLines: { width: 0 }
    };
    const tooltip = { enable: true };
    const chartarea = { border: { width: 0 } };
    const marker = { visible: false };
    chartLoad();
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} chartArea={chartarea}
title='Historical Indian Rupee Rate (INR USD)'>
        <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]}>
            <SeriesCollectionDirective>
                <SeriesDirective dataSource={series1} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
                    <TrendlinesDirective>
                        <TrendlineDirective type='Linear' fill='red' width={3}
marker={marker}>
                            </TrendlineDirective>
                        </TrendlinesDirective>
                    </SeriesDirective>
                </SeriesCollectionDirective>
            </ChartComponent>;
        </Inject>
    </ChartComponent>;
}
```

```
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
    ChartComponent, SeriesCollectionDirective, SeriesDirective,
    TrendlineDirective, TrendlinesDirective, Inject,
    Tooltip, LineSeries, ScatterSeries, SplineSeries, Trendlines, Category,
    ChartTheme, AxisModel, TooltipSettingsModel, ChartAreaModel
} from '@syncfusion/ej2-react-charts';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import { NumericTextBoxComponent } from '@syncfusion/ej2-react-inputs';
import { SampleBase } from '../common/sample-base';
import { PropertyPane } from '../common/property-pane';
import { EmitType } from '@syncfusion/ej2-base';
function App() {
    const series1: Object[] = [];
    const yValue: number[] = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89,
        8.68, 9.48, 10.11, 11.36, 12.34, 12.60, 12.95,
        13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52, 36.36,
        41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17, 41.20,
        43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
    function chartLoad(): void {
        let i: number;
        let j: number = 0;
        for (i = 1973; i <= 2013; i++) {
            series1.push({ x: i, y: yValue[j] });
            j++;
        }
    }
    const primaryxAxis: AxisModel = { title: 'Months', majorGridLines: {
width: 0 } };
    const primaryyAxis: AxisModel = {
        title: 'Rupees against Dollars', interval: 10,
        lineStyle: { width: 0 }, majorTickLines: { width: 0 }
    };
    const tooltip: TooltipSettingsModel = { enable: true };
    const chartarea: ChartAreaModel = { border: { width: 0 } };
    const marker = { visible: false };
    chartLoad();
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        tooltip={tooltip}
        chartArea={chartarea}
        title='Historical Indian Rupee Rate (INR USD)'>
        <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
        LineSeries, Trendlines]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={series1} xName='x' yName='y'
            name='Apple Inc' type='Scatter' fill='#0066FF'>
                <TrendlinesDirective>
```



```

        <TrendlineDirective type='Linear' fill='red' width={3}
marker={marker}>
        </TrendlineDirective>
    </TrendlinesDirective>
    </SeriesDirective>
</SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Forecasting

Trendline forecasting is the prediction of future/past situations.

Forward Forecasting and Backward Forecasting are the two types of forecasting.

Forward Forecasting

The value set for forwardForecast is used to determine the distance moving towards the future trend.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective,
TrendlineDirective, TrendlinesDirective, Inject, Tooltip, LineSeries,
ScatterSeries, SplineSeries, Trendlines, Category } from '@syncfusion/ej2-
react-charts';
function App() {
    const series1 = [];
    const yValue = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89, 8.68,
9.48, 10.11, 11.36, 12.34, 12.60, 12.95,
13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52,
36.36,
41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17,
41.20, 43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
    function chartLoad() {
        let i;
        let j = 0;
        for (i = 1973; i <= 2013; i++) {
            series1.push({ x: i, y: yValue[j] });
            j++;
        }
    }
    const primaryxAxis = { title: 'Months', majorGridLines: { width: 0 } };
    const primaryyAxis = {
        title: 'Rupees against Dollars', interval: 10,
        lineStyle: { width: 0 }, majorTickLines: { width: 0 }
    };
    const tooltip = { enable: true };
    const chartarea = { border: { width: 0 } };
    const marker = { visible: false };
    chartLoad();
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} chartArea={chartarea}
title='Historical Indian Rupee Rate (INR USD)'>

```

```

    <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={series1} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
        <TrendlinesDirective>
          <TrendlineDirective type='Linear' backwardForecast={5} width={3}
marker={marker}>>
        </TrendlineDirective>
      </TrendlinesDirective>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
  ChartComponent, SeriesCollectionDirective, SeriesDirective,
  TrendlineDirective, TrendlinesDirective, Inject,
  Tooltip, LineSeries, ScatterSeries, SplineSeries, Trendlines, Category,
  ChartTheme, AxisModel, TooltipSettingsModel, ChartAreaModel
} from '@syncfusion/ej2-react-charts';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import { NumericTextBoxComponent } from '@syncfusion/ej2-react-inputs';
import { SampleBase } from '../common/sample-base';
import { PropertyPane } from '../common/property-pane';
import { EmitType } from '@syncfusion/ej2-base';
function App() {
  const series1: Object[] = [];
  const yValue: number[] = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89,
8.68, 9.48, 10.11, 11.36, 12.34, 12.60, 12.95,
13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52, 36.36,
41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17, 41.20,
43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
  function chartLoad(): void {
    let i: number;
    let j: number = 0;
    for (i = 1973; i <= 2013; i++) {
      series1.push({ x: i, y: yValue[j] });
      j++;
    }
  }
  const primaryxAxis: AxisModel = { title: 'Months', majorGridLines: {
width: 0 } };
  const primaryyAxis: AxisModel = {
    title: 'Rupees against Dollars', interval: 10,
    lineStyle: { width: 0 }, majorTickLines: { width: 0 }
  };
  const tooltip: TooltipSettingsModel = { enable: true };

```

```

const chartarea: ChartAreaModel = { border: { width: 0 } };
const marker = { visible: false };
chartLoad();
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  tooltip={tooltip}
  chartArea={chartarea}
  title='Historical Indian Rupee Rate (INR USD)'>
  <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={series1} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
      <TrendlinesDirective>
        <TrendlineDirective type='Linear' backwardForecast={5} width={3}
marker={marker}>
        </TrendlineDirective>
      </TrendlinesDirective>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Backward Forecasting

The value set for the backwardForecast is used to determine the past trends.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective,
TrendlineDirective, TrendlinesDirective, Inject, Tooltip, LineSeries,
ScatterSeries, SplineSeries, Trendlines, Category } from '@syncfusion/ej2-
react-charts';
function App() {
  const series1 = [];
  const yValue = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89, 8.68,
9.48, 10.11, 11.36, 12.34, 12.60, 12.95,
13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52,
36.36,
41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17,
41.20, 43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
  function chartLoad() {
    let i;
    let j = 0;
    for (i = 1973; i <= 2013; i++) {
      series1.push({ x: i, y: yValue[j] });
      j++;
    }
  }
  const primaryxAxis = { title: 'Months', majorGridLines: { width: 0 } };
  const primaryyAxis = {
    title: 'Rupees against Dollars', interval: 10,

```

```

        lineStyle: { width: 0 }, majorTickLines: { width: 0 }
    };
    const tooltip = { enable: true };
    const chartarea = { border: { width: 0 } };
    const marker = { visible: false };
    chartLoad();
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} tooltip={tooltip} chartArea={chartarea}
title='Historical Indian Rupee Rate (INR USD)'>
    <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]}/>
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={series1} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
            <TrendlinesDirective>
                <TrendlineDirective type='Linear' backwardForecast={5} width={3}
marker={marker}>
            </TrendlineDirective>
        </TrendlinesDirective>
    </SeriesDirective>
</SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
    ChartComponent, SeriesCollectionDirective, SeriesDirective,
    TrendlineDirective, TrendlinesDirective, Inject,
    Tooltip, LineSeries, ScatterSeries, SplineSeries, Trendlines, Category,
    ChartTheme, AxisModel, TooltipSettingsModel, ChartAreaModel
} from '@syncfusion/ej2-react-charts';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import { NumericTextBoxComponent } from '@syncfusion/ej2-react-inputs';
import { SampleBase } from '../common/sample-base';
import { PropertyPane } from '../common/property-pane';
import { EmitType } from '@syncfusion/ej2-base';
function App() {
    const series1: Object[] = [];
    const yValue: number[] = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89,
8.68, 9.48, 10.11, 11.36, 12.34, 12.60, 12.95,
13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52, 36.36,
41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17, 41.20,
43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
    function chartLoad(): void {
        let i: number;
        let j: number = 0;
        for (i = 1973; i <= 2013; i++) {
            series1.push({ x: i, y: yValue[j] });
            j++;
        }
    }
}

```

```

    }
  }
  const primaryxAxis: AxisModel = { title: 'Months', majorGridLines: {
width: 0 } };
  const primaryyAxis: AxisModel = {
    title: 'Rupees against Dollars', interval: 10,
    lineStyle: { width: 0 }, majorTickLines: { width: 0 }
  };
  const tooltip: TooltipSettingsModel = { enable: true };
  const chartarea: ChartAreaModel = { border: { width: 0 } };
  const marker = { visible: false };
  chartLoad();
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    tooltip={tooltip}
    chartArea={chartarea}
    title='Historical Indian Rupee Rate (INR USD)'>
    <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={series1} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
        <TrendlinesDirective>
          <TrendlineDirective type='Linear' backwardForecast={5} width={3}
marker={marker}>
            </TrendlineDirective>
          </TrendlinesDirective>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  );
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Show or hide a trendline

You can show or hide the trendline by setting trendline **visible** property.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective,
TrendlineDirective, TrendlinesDirective, Inject, Tooltip, LineSeries,
ScatterSeries, SplineSeries, Trendlines, Category } from '@syncfusion/ej2-
react-charts';
function App() {
    const series1 = [];
    const yValue = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89, 8.68,
9.48, 10.11, 11.36, 12.34, 12.60, 12.95,
13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52,
36.36,
41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17,
41.20, 43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
    function chartLoad() {
        let i;
```

```

        let j = 0;
        for (i = 1973; i <= 2013; i++) {
            series1.push({ x: i, y: yValue[j] });
            j++;
        }
    }
    const primaryxAxis = { title: 'Months', majorGridLines: { width: 0 } };
    const primaryyAxis = {
        title: 'Rupees against Dollars', interval: 10,
        lineStyle: { width: 0 }, majorTickLines: { width: 0 }
    };
    const tooltip = { enable: true };
    const chartarea = { border: { width: 0 } };
    const marker = { visible: false };
    chartLoad();
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis} tooltip={tooltip} chartArea={chartarea}
    title='Historical Indian Rupee Rate (INR USD)'>
        <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
    LineSeries, Trendlines]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={series1} xName='x' yName='y'
    name='Apple Inc' type='Scatter' fill='#0066FF'>
                <TrendlinesDirective>
                    <TrendlineDirective type='Linear' visible={false}>
                    </TrendlineDirective>
                </TrendlinesDirective>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { Browser } from '@syncfusion/ej2-base';
import {
    ChartComponent, SeriesCollectionDirective, SeriesDirective,
    TrendlineDirective, TrendlinesDirective, Inject,
    Tooltip, LineSeries, ScatterSeries, SplineSeries, Trendlines, Category,
    ChartTheme
    ,AxisModel,TooltipSettingsModel,ChartAreaModel
} from '@syncfusion/ej2-react-charts';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import { NumericTextBoxComponent } from '@syncfusion/ej2-react-inputs';
import { SampleBase } from '../common/sample-base';
import { PropertyPane } from '../common/property-pane';
import { EmitType } from '@syncfusion/ej2-base';
function App() {
    const series1: Object[] = [];
    const yValue: number[] = [7.66, 8.03, 8.41, 8.97, 8.77, 8.20, 8.16, 7.89,
    8.68, 9.48, 10.11, 11.36, 12.34, 12.60, 12.95,

```

```

    13.91, 16.21, 17.50, 22.72, 28.14, 31.26, 31.39, 32.43, 35.52, 36.36,
    41.33, 43.12, 45.00, 47.23, 48.62, 46.60, 45.28, 44.01, 45.17, 41.20,
    43.41, 48.32, 45.65, 46.61, 53.34, 58.53];
    function chartLoad(): void {
        let i: number;
        let j: number = 0;
        for (i = 1973; i <= 2013; i++) {
            series1.push({ x: i, y: yValue[j] });
            j++;
        }
    }
    const primaryxAxis: AxisModel = { title: 'Months', majorGridLines: {
width: 0 } };
    const primaryyAxis: AxisModel = {
        title: 'Rupees against Dollars', interval: 10,
        lineStyle: { width: 0 }, majorTickLines: { width: 0 }
    };
    const tooltip: TooltipSettingsModel = { enable: true };
    const chartarea: ChartAreaModel = { border: { width: 0 } };
    const marker = { visible: false };
    chartLoad();
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        tooltip={tooltip}
        chartArea={chartarea}
        title='Historical Indian Rupee Rate (INR USD)'>
        <Inject services={[Category, Tooltip, ScatterSeries, SplineSeries,
LineSeries, Trendlines]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={series1} xName='x' yName='y'
name='Apple Inc' type='Scatter' fill='#0066FF'>
                <TrendlinesDirective>
                    <TrendlineDirective type='Linear' visible={false} >
                        </TrendlineDirective>
                    </TrendlinesDirective>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>
    </>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Data markers in React Chart component

Data markers are used to provide information about the data points in the series. You can add a shape to adorn each data point.

<!-- markdownlint-disable MD036 -->

Marker

<!-- markdownlint-disable MD036 -->

Markers can be added to points by enabling the [visible](#) option of the marker property. By default, distinct markers will be enabled for each series in the chart.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'DateTime' };
  const marker = { visible: true };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' type='Line'
marker={marker}>
    </SeriesDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y1' type='Line'
marker={marker}>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
  AxisModel, ChartComponent, SeriesCollectionDirective, SeriesDirective,
  Inject,
  Legend, DateTime, Tooltip, DataLabel, LineSeries
}
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime' };
  const marker = { visible: true };
  const data: any[] = [
    { x: new Date(2016, 0, 1), y: -7.1, y1: -3.5 }, { x: new Date(2016, 1,
1), y: -3.7, y1: 0 },
    { x: new Date(2016, 2, 1), y: 0.8, y1: 5 }, { x: new Date(2016, 3, 1),
y: 6.3, y1: 10.3 },
    { x: new Date(2016, 4, 1), y: 13.3, y1: 18.3 }, { x: new Date(2016, 5,
1), y: 18.0, y1: 21 },
    { x: new Date(2016, 6, 1), y: 19.8, y1: 23.5 }, { x: new Date(2016, 7,
1), y: 18.1, y1: 21.3 },
    { x: new Date(2016, 8, 1), y: 13.1, y1: 16.3 }, { x: new Date(2016, 9,
1), y: 4.1, y1: 9 },
    { x: new Date(2016, 10, 1), y: -3.8, y1: -1.8 }, { x: new Date(2016, 11,
1), y: -6.8, y1: -3.8 }
  ];
}
```



```

];
return <ChartComponent id='charts'
  primaryXAxis={primaryXAxis}>
  <Inject services={[LineSeries, Legend, Tooltip, DataLabel, DateTime]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y'
      type='Line' marker={marker}>
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y1'
      type='Line' marker={marker}>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Shape

Markers can be assigned with different shapes such as Rectangle, Circle, Diamond, etc. using the [shape](#) property.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
import { data } from 'datasource.ts';
function App() {
  const primaryXAxis = { valueType: 'DateTime' };
  const marker = { visible: true, height: 10, width: 10, shape: 'Pentagon'
};
  return <ChartComponent id='charts' primaryXAxis={primaryXAxis}>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel,
DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='Warmest' type='Line' marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
  Legend, DateTime, Tooltip, DataLabel, LineSeries}

```

```

from '@syncfusion/ej2-react-charts';
function App() {
  const primaryXAxis: AxisModel = { valueType: 'DateTime' };
  const marker = { visible: true, height: 10, width: 10, shape: 'Pentagon'
};
  const data: any[] = [
    { x: new Date(2016, 0, 1), y: -7.1 }, { x: new Date(2016, 1, 1), y: -3.7
},
    { x: new Date(2016, 2, 1), y: 0.8 }, { x: new Date(2016, 3, 1), y: 6.3
},
    { x: new Date(2016, 4, 1), y: 13.3 }, { x: new Date(2016, 5, 1), y: 18.0
},
    { x: new Date(2016, 6, 1), y: 19.8 }, { x: new Date(2016, 7, 1), y: 18.1
},
    { x: new Date(2016, 8, 1), y: 13.1 }, { x: new Date(2016, 9, 1), y: 4.1
},
    { x: new Date(2016, 10, 1), y: -3.8 }, { x: new Date(2016, 11, 1), y: -
6.8 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryXAxis}>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, DateTime]}
  />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='Warmest'
      type='Line' marker={marker}>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Note : To know more about the marker shape type refer the [shape](#).

Images

Apart from the shapes, you can also add custom images to mark the data point using the [imageUrl](#) property.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
function App() {
  const data = [
    { x: "Jan", y: 60 }, { x: "Feb", y: 50 },
    { x: "Mar", y: 64 }, { x: "Apr", y: 63 },
    { x: "May", y: 81 }, { x: "Jun", y: 64 },
    { x: "Jul", y: 82 }, { x: "Aug", y: 96 },
    { x: "Sep", y: 78 }, { x: "Oct", y: 60 },

```

```

    { x: "Nov", y: 58 }, { x: "Dec", y: 56 }
  ];
  const primaryxAxis = { title: 'Month', valueType: 'Category' };
  const primaryyAxis = {
    title: 'Temperature (°C)', rangePadding: 'None', labelFormat:
    '{value}°C',
    minimum: 0, maximum: 100
  };
  const marker = {
    visible: true, width: 10, height: 10, shape: 'Image',
    imageUrl: './sun_annotation.png'
  };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Temperature flow over months'>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India' type='Line' marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
  AxisModel, ChartComponent, SeriesCollectionDirective, SeriesDirective,
  Inject,
  Legend, Category, Tooltip, DataLabel, LineSeries, MarkerSettingsModel
} from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { x: "Jan", y: 60 }, { x: "Feb", y: 50 },
    { x: "Mar", y: 64 }, { x: "Apr", y: 63 },
    { x: "May", y: 81 }, { x: "Jun", y: 64 },
    { x: "Jul", y: 82 }, { x: "Aug", y: 96 },
    { x: "Sep", y: 78 }, { x: "Oct", y: 60 },
    { x: "Nov", y: 58 }, { x: "Dec", y: 56 }
  ];
  const primaryxAxis: AxisModel = { title: 'Month', valueType: 'Category' };
  const primaryyAxis: AxisModel = {
    title: 'Temperature (°C)', rangePadding: 'None', labelFormat:
    '{value}°C',
    minimum: 0, maximum: 100
  };
  const marker: MarkerSettingsModel = {
    visible: true, width: 10, height: 10, shape: 'Image',
    imageUrl: './sun_annotation.png'
  };
}

```

```

return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  title='Temperature flow over months'>
  <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='India'
    type='Line' marker={marker}>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% enddraw %}

```

Customization

Marker's color and border can be customized using **fill** and **border** properties.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'DateTime' };
  const marker = {
    visible: true,
    fill: 'Red', height: 10, width: 10,
    border: { width: 2, color: 'blue' },
  };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel,
DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='Warmest' type='Line' marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,

```

```

    Legend, DateTime, Tooltip, DataLabel, LineSeries}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime' };
    const marker = {
        visible: true,
        fill: 'Red', height: 10, width: 10,
        border: { width: 2, color: 'blue' },
    };
    const data: any[] = [
        { x: new Date(2016, 0, 1), y: -7.1 }, { x: new Date(2016, 1, 1), y: -3.7 },
        { x: new Date(2016, 2, 1), y: 0.8 }, { x: new Date(2016, 3, 1), y: 6.3 },
        { x: new Date(2016, 4, 1), y: 13.3 }, { x: new Date(2016, 5, 1), y: 18.0 },
        { x: new Date(2016, 6, 1), y: 19.8 }, { x: new Date(2016, 7, 1), y: 18.1 },
        { x: new Date(2016, 8, 1), y: 13.1 }, { x: new Date(2016, 9, 1), y: 4.1 },
        { x: new Date(2016, 10, 1), y: -3.8 }, { x: new Date(2016, 11, 1), y: -6.8 }
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel, DateTime]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
            name='Warmest'
                type='Line' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Customizing specific point

You can also customize the specific marker and label using [pointRender](#) event. The `pointRender` event allows you to change the shape, color and border for a point.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
import { data } from 'datasource.ts';
function App() {
    const pointRender = (args) => {
        if (args.point.index === 3) {
            args.fill = 'red';
        }
    };
};

```

```

const primaryxAxis = { valueType: 'DateTime' };
const marker = { visible: true, height: 10, width: 10 };
return <ChartComponent id='charts' pointRender={pointRender}
primaryXAxis={primaryxAxis}>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel,
DateTime]}/>
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='Warmest' type='Line' marker={marker}>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, IPointRenderEventArgs,
Legend, DateTime, Tooltip, DataLabel, LineSeries }
from '@syncfusion/ej2-react-charts';
function App() {
    const pointRender = (args: IPointRenderEventArgs) => {
        if (args.point.index === 3) {
            args.fill = 'red'
        }
    };
    const primaryxAxis: AxisModel = { valueType: 'DateTime' };
    const marker = { visible: true, height: 10, width: 10 };
    const data: any[] = [
        { x: new Date(2016, 0, 1), y: -7.1 }, { x: new Date(2016, 1, 1), y: -3.7
    },
        { x: new Date(2016, 2, 1), y: 0.8 }, { x: new Date(2016, 3, 1), y: 6.3
    },
        { x: new Date(2016, 4, 1), y: 13.3 }, { x: new Date(2016, 5, 1), y: 18.0
    },
        { x: new Date(2016, 6, 1), y: 19.8 }, { x: new Date(2016, 7, 1), y: 18.1
    },
        { x: new Date(2016, 8, 1), y: 13.1 }, { x: new Date(2016, 9, 1), y: 4.1
    },
        { x: new Date(2016, 10, 1), y: -3.8 }, { x: new Date(2016, 11, 1), y: -
6.8 }
    ];
    return <ChartComponent id='charts' pointRender={pointRender}
primaryXAxis={primaryxAxis}>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel, DateTime]}
/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='Warmest'
            type='Line' marker={marker}>
            </SeriesDirective>

```

```

        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Fill marker with series color

Marker can be filled with the series color by setting the [isFilled](#) property to **true**.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
import { data } from 'datasource.ts';
class App extends React.Component {
    primaryxAxis = { valueType: 'DateTime' };
    marker = { visible: true, height: 10, width: 10, isFilled: true };
    render() {
        return <ChartComponent id='charts' pointRender={this.pointRender}
primaryXAxis={this.primaryxAxis}>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel,
DateTime]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='Warmest' type='Line' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
    }
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, IPointRenderEventArgs,
Legend, DateTime, Tooltip, DataLabel, LineSeries}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime' };
    const marker = { visible: true, height: 10, width: 10, isFilled: true };
    const data: any[] = [
        { x: new Date(2016, 0, 1), y: -7.1 }, { x: new Date(2016, 1, 1), y: -3.7
    },
        { x: new Date(2016, 2, 1), y: 0.8 }, { x: new Date(2016, 3, 1), y: 6.3
    },
        { x: new Date(2016, 4, 1), y: 13.3 }, { x: new Date(2016, 5, 1), y: 18.0
    },
    ],
}

```

```

    { x: new Date(2016, 6, 1), y: 19.8 }, { x: new Date(2016, 7, 1), y: 18.1
  },
  { x: new Date(2016, 8, 1), y: 13.1 }, { x: new Date(2016, 9, 1), y: 4.1
  },
  { x: new Date(2016, 10, 1), y: -3.8 }, { x: new Date(2016, 11, 1), y: -
6.8 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, DateTime]}
  />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='Warmest'
        type='Line' marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

See Also

- [Customize the marker with different shape](#)

Data labels in React Chart component

Data label can be added to a chart series by enabling the [visible](#) option in the dataLabel. By default, the labels will arrange smartly without overlapping.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'DateTime' };
  const marker = {
    visible: true,
    height: 10, width: 10,
    shape: 'Pentagon',
    dataLabel: { visible: true }
  };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel,
DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='Warmest' type='Line' marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
ReactDOM.render(<App />, document.getElementById("charts"));

```



```

    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
  AxisModel, ChartComponent, SeriesCollectionDirective, SeriesDirective,
  Inject,
  Legend, DateTime, Tooltip, DataLabel, LineSeries, MarkerSettingsModel
} from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime' };
  const marker: MarkerSettingsModel = {
    visible: true,
    height: 10, width: 10,
    shape: 'Pentagon',
    dataLabel: { visible: true }
  };
  const data: any[] = [
    { x: new Date(2016, 0, 1), y: -7.1 }, { x: new Date(2016, 1, 1), y: -3.7
  },
    { x: new Date(2016, 2, 1), y: 0.8 }, { x: new Date(2016, 3, 1), y: 6.3
  },
    { x: new Date(2016, 4, 1), y: 13.3 }, { x: new Date(2016, 5, 1), y: 18.0
  },
    { x: new Date(2016, 6, 1), y: 19.8 }, { x: new Date(2016, 7, 1), y: 18.1
  },
    { x: new Date(2016, 8, 1), y: 13.1 }, { x: new Date(2016, 9, 1), y: 4.1
  },
    { x: new Date(2016, 10, 1), y: -3.8 }, { x: new Date(2016, 11, 1), y: -
6.8 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='Warmest'
        type='Line' marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Note: To use data label feature, we need to inject **DataLabel** module into the **services**.

Position

Using [position](#) property, you can place the label either on **Top**, **Middle**, **Bottom** or **Outer** (outer is applicable for column and bar type series).

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
import { data } from 'datasource.ts';
function App() {
    const primaryXAxis = { title: 'Months', valueType: 'Category' };
    const marker = { dataLabel: { visible: true, position: 'Middle' } };
    return <ChartComponent id='charts' primaryXAxis={primaryXAxis}>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='Warmest' type='Line' marker={marker}>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
    }
    ;
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, LineSeries}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryXAxis: AxisModel = { title: 'Months', valueType: 'Category'
};
    const marker = { dataLabel: { visible: true, position: 'Middle' } };
    const data: any[] = [
        { x: new Date(2016, 0, 1), y: -7.1 }, { x: new Date(2016, 1, 1), y: -3.7
},
        { x: new Date(2016, 2, 1), y: 0.8 }, { x: new Date(2016, 3, 1), y: 6.3
},
        { x: new Date(2016, 4, 1), y: 13.3 }, { x: new Date(2016, 5, 1), y: 18.0
},
        { x: new Date(2016, 6, 1), y: 19.8 }, { x: new Date(2016, 7, 1), y: 18.1
},
        { x: new Date(2016, 8, 1), y: 13.1 }, { x: new Date(2016, 9, 1), y: 4.1
},
        { x: new Date(2016, 10, 1), y: -3.8 }, { x: new Date(2016, 11, 1), y: -
6.8 }
    ];
    return <ChartComponent id='charts'
```

```

    primaryXAxis={primaryxAxis}>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category]}
  />

    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='Warmest'
      type='Line' marker={marker}>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Note: The position **Outer** is applicable for column and bar type series.

Data Label Template

Label content can be formatted by using the template option. Inside the template, you can add the placeholder text `${point.x}` and `${point.y}` to display corresponding data points x & y value. Using [template](#) property, you can set data label template in chart.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
function App() {
  const primaryxAxis = { title: 'Months', valueType: 'DateTime',
intervalType: 'Months' };
  const template = chartTemplate;
  const marker = { dataLabel: { visible: true, position: 'Middle', template:
template } };
  function chartTemplate(args) {
    return (<div id="templateWrap" style={{ border: '1px solid black',
backgroundColor: 'red', padding: '3px 3px 3px 3px' }}>
      <div>{args.point.x.getFullYear()}</div>
      <div>{args.point.y}</div>
    </div>);
  }
  ;
  const data = [
    { x: new Date(2016, 0, 1), y: -7.1 }, { x: new Date(2016, 1, 1), y: -3.7
},
    { x: new Date(2016, 2, 1), y: 0.8 }, { x: new Date(2016, 3, 1), y: 6.3
},
    { x: new Date(2016, 4, 1), y: 13.3 }, { x: new Date(2016, 5, 1), y: 18.0
},
    { x: new Date(2016, 6, 1), y: 19.8 }, { x: new Date(2016, 7, 1), y: 18.1
},
    { x: new Date(2016, 8, 1), y: 13.1 }, { x: new Date(2016, 9, 1), y: 4.1
},
    { x: new Date(2016, 10, 1), y: -3.8 }, { x: new Date(2016, 11, 1), y: -
6.8 }
  ];

```

```

];
return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
  <Inject services={[LineSeries, Legend, Tooltip, DataLabel, DateTime]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='Warmest' type='Line' marker={marker}>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
  AxisModel, ChartComponent, SeriesCollectionDirective, SeriesDirective,
  Inject,
  Legend, DateTime, Tooltip, DataLabel, LineSeries, MarkerSettingsModel
}
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { title: 'Months', valueType: 'DateTime',
intervalType: 'Months' };
  const template: any = chartTemplate;
  const marker: MarkerSettingsModel = { dataLabel: { visible: true,
position: 'Middle', template: template } };
  function chartTemplate(args: any) {
    return (<div id="templateWrap" style={{ border: '1px solid black',
backgroundColor: 'red', padding: '3px 3px 3px 3px' }}>
      <div>{args.point.x.getFullYear()}</div>
      <div>{args.point.y}</div>
    </div>);
  };
  const data: any[] = [
    { x: new Date(2016, 0, 1), y: -7.1 }, { x: new Date(2016, 1, 1), y: -3.7
},
    { x: new Date(2016, 2, 1), y: 0.8 }, { x: new Date(2016, 3, 1), y: 6.3
},
    { x: new Date(2016, 4, 1), y: 13.3 }, { x: new Date(2016, 5, 1), y: 18.0
},
    { x: new Date(2016, 6, 1), y: 19.8 }, { x: new Date(2016, 7, 1), y: 18.1
},
    { x: new Date(2016, 8, 1), y: 13.1 }, { x: new Date(2016, 9, 1), y: 4.1
},
    { x: new Date(2016, 10, 1), y: -3.8 }, { x: new Date(2016, 11, 1), y: -
6.8 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, DateTime]} />

```

```

    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='Warmest'
      type='Line' marker={marker}>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Text Mapping

Text from the data source can be mapped using `name` property.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
import { mapData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'Category' };
  const marker = { dataLabel: { visible: true, name: 'text' } };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={mapData} xName='x' yName='y'
type='Column' marker={marker}>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries}
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'Category' };
  const marker = { dataLabel: { visible: true, name: 'text' } };
  const mapData: any[] = [{ x: 'Jan', y: 12, text: 'January : 12°C' }, { x:
'Feb', y: 8, text: 'February : 8°C' },
{ x: 'Mar', y: 11, text: 'March : 11°C' }, { x: 'Apr', y: 6, text: 'April
: 6°C' }];

```

```

return <ChartComponent id='charts'
  primaryXAxis={primaryXAxis}>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={mapData} xName='x' yName='y'
      type='Column' marker={marker}>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Format

Data label for the chart can be formatted using [format](#) property. You can use the global formatting options, such as 'n', 'p', and 'c'.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
import { mapData } from 'datasource.ts';
class App extends React.Component {
  primaryXAxis = { valueType: 'Category' };
  marker = { dataLabel: { visible: true, format: 'n2' } };
  render() {
    return <ChartComponent id='charts' primaryXAxis={this.primaryXAxis}>
      <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={mapData} xName='x' yName='y'
          type='Column' marker={this.marker}>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
  }
}
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
  Legend, Category, Tooltip, DataLabel, ColumnSeries}
from '@syncfusion/ej2-react-charts';
class App extends React.Component<{}, {}> {
  public primaryXAxis: AxisModel = { valueType: 'Category' };

```

```

    public marker = { dataLabel: { visible: true, format: 'n2' } };
    public mapData: any[] = [{ x: 'Jan', y: 12, text: 'January : 12°C' }, { x:
'Feb', y: 8, text: 'February : 8°C' },
    { x: 'Mar', y: 11, text: 'March : 11°C' }, { x: 'Apr', y: 6, text:
'April : 6°C' }];
    render() {
        return <ChartComponent id='charts'
            primaryXAxis={this.primaryxAxis}>
            <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
            <SeriesCollectionDirective>
                <SeriesDirective dataSource={this.mapData} xName='x' yName='y'
                    type='Column' marker={this.marker}>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>
    }
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Value	Format	Resultant Value	Description
1000	n1	1000.0	The number is rounded to 1 decimal place.
1000	n2	1000.00	The number is rounded to 2 decimal places.
1000	n3	1000.000	The number is rounded to 3 decimal place.
0.01	p1	1.0%	The number is converted to percentage with 1 decimal place.
0.01	p2	1.00%	The number is converted to percentage with 2 decimal place.
0.01	p3	1.000%	The number is converted to percentage with 3 decimal place.
1000	c1	\$1000.0	The currency symbol is appended to number and number is rounded to 1 decimal place.
1000	c2	\$1000.00	The currency symbol is appended to number and number is rounded to 2 decimal place.

Margin

margin for data label can be applied to using left, right, bottom and top properties.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
import { columnData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Category' };
    const marker = {

```

```

        dataLabel: {
            visible: true, border: { width: 1, color: 'red' },
            margin: {
                left: 5,
                right: 5,
                top: 5,
                bottom: 5
            }
        }
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={columnData} xName='country'
yName='gold' type='Column' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
        Legend, Category, Tooltip, DataLabel, ColumnSeries}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Category' };
    const marker = {
        dataLabel: {
            visible: true, border: { width: 1, color: 'red' },
            margin: {
                left: 5,
                right: 5,
                top: 5,
                bottom: 5
            }
        }
    };
    const columnData: any[] = [
        { country: "USA", gold: 50 },
        { country: "China", gold: 40 },
        { country: "Japan", gold: 70 },
        { country: "Australia", gold: 60 },
        { country: "France", gold: 50 },
        { country: "Germany", gold: 40 },
        { country: "Italy", gold: 40 },
        { country: "Sweden", gold: 30 }
    ];
}

```



```

return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={columnData} xName='country'
yName='gold'
            type='Column' marker={marker}>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

DataLabel Rotation

Using **angle** property, you can rotate the data label by its given angle.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
import { columnData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Category' };
    const marker = {
        //Data label angle as 45
        dataLabel: {
            visible: true, border: { width: 1, color: 'red' },
            margin: {
                left: 5,
                right: 5,
                top: 5,
                bottom: 5
            }, angle: 45, enableRotation: true
        }
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={columnData} xName='country'
yName='gold' type='Column' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
      Legend, Category, Tooltip, DataLabel, ColumnSeries }
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'Category' };
  const marker = {
    //Data label angle as 45
    dataLabel: {
      visible: true, border: { width: 1, color: 'red' },
      margin: {
        left: 5,
        right: 5,
        top: 5,
        bottom: 5
      }, angle: 45, enableRotation: true
    }
  };
  const columnData: any[] = [
    { country: "USA", gold: 50 },
    { country: "China", gold: 40 },
    { country: "Japan", gold: 70 },
    { country: "Australia", gold: 60 },
    { country: "France", gold: 50 },
    { country: "Germany", gold: 40 },
    { country: "Italy", gold: 40 },
    { country: "Sweden", gold: 30 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={columnData} xName='country'
yName='gold'
        type='Column' marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Customization

stroke and border of data label can be customized using fill and border properties. Rounded corners can be customized using rx and ry properties. The data label can be customized by using the color, fontFamily, fontWeight, size properties in the font property of the data label.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";

```

```

import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
import { columnData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Category' };
    const marker = {
        dataLabel: {
            visible: true, font: { color: "blue", fontWeight: "500" },
border: { width: 2, color: 'red' },
            rx: 10, ry: 10
        }
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={columnData} xName='country'
yName='gold' type='Column' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Category' };
    const marker = {
        dataLabel: {
            visible: true, font: { color: "blue", fontWeight: "500" }, border: {
width: 2, color: 'red' },
            rx: 10, ry: 10
        }
    };
    const columnData: any[] = [
        { country: "USA", gold: 50 },
        { country: "China", gold: 40 },
        { country: "Japan", gold: 70 },
        { country: "Australia", gold: 60 },
        { country: "France", gold: 50 },
        { country: "Germany", gold: 40 },
        { country: "Italy", gold: 40 },
        { country: "Sweden", gold: 30 }
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}>

```

```

    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={columnData} xName='country'
yName='gold'
        type='Column' marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Note: **rx** and **ry** properties requires **border** values not to be null.

Customizing Specific Point

You can also customize the specific marker and label using [pointRender](#) and [textRender](#) event. **pointRender** event allows you to change the shape, color and border for a point, whereas the **textRender** event allows you to change the text for the point.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
import { data } from 'datasource.ts';
function App() {
  const pointRender = (args) => {
    if (args.point.index === 6) {
      args.fill = 'red';
    }
  };
  const primaryxAxis = { valueType: 'Category' };
  const marker = { dataLabel: { visible: true, position: 'Middle' } };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
pointRender={pointRender} title='Alaska Weather Statistics - 2016'>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' type='Line'
marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";

```

```

import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
        Legend, Category, Tooltip, DataLabel, LineSeries}
from '@syncfusion/ej2-react-charts';
function App() {
    const pointRender: EmitType<IPointRenderEventArgs> = (args:
IPointRenderEventArgs): void => {
        if (args.point.index === 6) {
            args.fill = 'red'
        }
    };
    const primaryxAxis: AxisModel = { valueType: 'Category' };
    const marker = { dataLabel: { visible: true, position: 'Middle' } };
    const data: any[] = [
        { x: new Date(2016, 0, 1), y: -7.1 }, { x: new Date(2016, 1, 1), y: -3.7
    },
        { x: new Date(2016, 2, 1), y: 0.8 }, { x: new Date(2016, 3, 1), y: 6.3
    },
        { x: new Date(2016, 4, 1), y: 13.3 }, { x: new Date(2016, 5, 1), y: 18.0
    },
        { x: new Date(2016, 6, 1), y: 19.8 }, { x: new Date(2016, 7, 1), y: 18.1
    },
        { x: new Date(2016, 8, 1), y: 13.1 }, { x: new Date(2016, 9, 1), y: 4.1
    },
        { x: new Date(2016, 10, 1), y: -3.8 }, { x: new Date(2016, 11, 1), y: -
6.8 }
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        pointRender={pointRender}
        title='Alaska Weather Statistics - 2016'>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category]}
    />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y'
                type='Line' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Show percentage based on each series points

You can calculate the percentage value based on the sum for each series using the `seriesRender` and `textRender` events in the chart. In `seriesRender` calculate the sum of each series y values and In `textRender` calculate percentage value based on the sum value and modify the text.

INDEX.JSX

```

{% raw %}
import { render } from 'react-dom';
import * as React from "react";

```

```

import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, ColumnSeries, DataLabel } from '@syncfusion/ej2-
react-charts';
import { Browser } from '@syncfusion/ej2-base';
const data1 = [{ x: 'USA', y: 46 }, { x: 'GBR', y: 27 }, { x: 'CHN', y: 26
}];
const data2 = [{ x: 'USA', y: 37 }, { x: 'GBR', y: 23 }, { x: 'CHN', y: 18
}];
const data3 = [{ x: 'USA', y: 38 }, { x: 'GBR', y: 17 }, { x: 'CHN', y: 26
}];
const total = [];
function App() {
    return (<div className='control-pane'>
        <div className='control-section'>
            <ChartComponent id='charts' style={{ textAlign: "center" }}
textRender={onTextRender.bind(this)} primaryXAxis={{ valueType: 'Category',
interval: 1, majorGridLines: { width: 0 } }} primaryYAxis={{
            majorGridLines: { width: 0 },
            majorTickLines: { width: 0 }, lineStyle: { width: 0 },
labelStyle: { color: 'transparent' }
            }} chartArea={{ border: { width: 0 } }} tooltip={{ enable: true }}
width={Browser.isDevice ? '100%' : '60%'} title='Olympic Medal Counts - RIO'
seriesRender={onSeriesRender.bind(this)}>
                <Inject services={[ColumnSeries, Legend, Tooltip,
Category, DataLabel]}>
                    <SeriesCollectionDirective>
                        <SeriesDirective dataSource={data1} xName='x'
yName='y' name='Gold' type='Column' marker={{ dataLabel: { visible: true,
position: 'Top', font: { fontWeight: '600', color: '#ffffff' } } }}>
                        </SeriesDirective>
                        <SeriesDirective dataSource={data2} xName='x'
yName='y' name='Silver' type='Column' marker={{ dataLabel: { visible: true,
position: 'Top', font: { fontWeight: '600', color: '#ffffff' } } }}>
                        </SeriesDirective>
                        <SeriesDirective dataSource={data3} xName='x'
yName='y' name='Bronze' type='Column' marker={{ dataLabel: { visible: true,
position: 'Top', font: { fontWeight: '600', color: '#ffffff' } } }}>
                        </SeriesDirective>
                    </SeriesCollectionDirective>
                </ChartComponent>
            </div>
        </div>);
    function onSeriesRender(args) {
        for (var i = 0; i < args.data.length; i++) {
            if (!total[args.data[i].x])
                total[args.data[i].x] = 0;
            total[args.data[i].x] += parseInt(args.data[i].y);
        }
    }
    ;
    function onTextRender(args) {
        var percentage = (parseInt(args.text) / total[args.point.x]) * 100;
        percentage = percentage % 1 === 0 ? percentage :
percentage.toFixed(2);
        args.text = percentage + '%';
    }
    ;
}

```

```

}
export default App;
render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import { render } from 'react-dom';
import * as React from "react";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, ColumnSeries, DataLabel } from '@syncfusion/ej2-
react-charts';
import { Browser } from '@syncfusion/ej2-base';
const data1 = [{ x: 'USA', y: 46 }, { x: 'GBR', y: 27 }, { x: 'CHN', y: 26
}];
const data2 = [{ x: 'USA', y: 37 }, { x: 'GBR', y: 23 }, { x: 'CHN', y: 18
}];
const data3 = [{ x: 'USA', y: 38 }, { x: 'GBR', y: 17 }, { x: 'CHN', y: 26
}];
const total = [];
function App() {
    return (<div className='control-pane'>
        <div className='control-section'>
            <ChartComponent id='charts' style={{ textAlign: "center" }}
textRender={onTextRender.bind(this)} primaryXAxis={{ valueType: 'Category',
interval: 1, majorGridLines: { width: 0 } }} primaryYAxis={{
            majorGridLines: { width: 0 },
            majorTickLines: { width: 0 }, lineStyle: { width: 0 },
labelStyle: { color: 'transparent' }
            }} chartArea={{ border: { width: 0 } }} tooltip={{ enable:
true }} width={Browser.isDevice ? '100%' : '60%'} title='Olympic Medal
Counts - RIO' seriesRender={onSeriesRender.bind(this)}>
                <Inject services={[ColumnSeries, Legend, Tooltip,
Category, DataLabel]} />
                <SeriesCollectionDirective>
                    <SeriesDirective dataSource={data1} xName='x'
yName='y' name='Gold' type='Column' marker={{ dataLabel: { visible: true,
position: 'Top', font: { fontWeight: '600', color: '#ffffff' } } }}>
                        </SeriesDirective>
                    <SeriesDirective dataSource={data2} xName='x'
yName='y' name='Silver' type='Column' marker={{ dataLabel: { visible: true,
position: 'Top', font: { fontWeight: '600', color: '#ffffff' } } }}>
                        </SeriesDirective>
                    <SeriesDirective dataSource={data3} xName='x'
yName='y' name='Bronze' type='Column' marker={{ dataLabel: { visible: true,
position: 'Top', font: { fontWeight: '600', color: '#ffffff' } } }}>
                        </SeriesDirective>
                </SeriesCollectionDirective>
            </ChartComponent>
        </div>
    </div>);
    function onSeriesRender(args) {
        for (var i = 0; i < args.data.length; i++) {
            if (!total[args.data[i].x]) total[args.data[i].x] = 0;
            total[args.data[i].x] += parseInt(args.data[i].y);
        }
    }
}

```

```

    }
  }
;
function onTextRender(args) {
  var percentage = (parseInt(args.text) / total[args.point.x]) * 100;
  percentage = percentage % 1 === 0 ? percentage :
percentage.toFixed(2);
  args.text = percentage + '%';
};
}
export default App;
render(<App />, document.getElementById('charts'));
{% endraw %}

```

See Also

- [Show total stacking values in data label](#)
- [Prevent the data label when the data value is 0](#)

Chart annotations in React Chart component

Annotations are used to mark the specific area of interest in the chart area with texts, shapes or images.

<!-- markdownlint-disable MD033 -->

You can add annotations to the chart by using the `<code>annotations</code> option. By using the content option of annotation object, you can specify the id of the element that needs to be displayed in the chart area.`

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ChartAnnotation, AnnotationsDirective, AnnotationDirective, Legend,
Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-react-
charts';
import { columnData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'Category', title: 'Countries' };
  const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
  const border = { width: 2, color: 'grey' };
  const animation = { enable: true, duration: 1200, delay: 100 };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Olympic Medals'>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
ChartAnnotation]} />
  <AnnotationsDirective>
    <AnnotationDirective content='70 Gold Medals' region='Series'
coordinateUnits='Point' x='Japan' y={75}>
    </AnnotationDirective>
  </AnnotationsDirective>
  <SeriesCollectionDirective>

```



```

        <SeriesDirective dataSource={columnData} xName='country'
yName='gold' name='Gold' border={border} animation={animation}
type='Column'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, ChartAnnotation, AnnotationsDirective,
AnnotationDirective, Legend, Category, Tooltip, DataLabel,
ColumnSeries } from '@syncfusion/ej2-react-charts';
import { columnData } from 'datasource.ts';
function App() {
    const primaryXAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
    const primaryYAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
    const border = { width: 2, color: 'grey' };
    const animation = { enable: true, duration: 1200, delay: 100 };
    return <ChartComponent id='charts'
        primaryXAxis={primaryXAxis}
        primaryYAxis={primaryYAxis}
        title='Olympic Medals'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
ChartAnnotation]} />
        <AnnotationsDirective>
            <AnnotationDirective content='70 Gold Medals' region='Series'
coordinateUnits='Point' x='Japan' y={75}>
            </AnnotationDirective>
        </AnnotationsDirective>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={columnData} xName='country'
yName='gold' name='Gold'
                border={border}
                animation={animation} type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Note: To use annotation feature in chart, we need to inject **ChartAnnotation** module into the **services**.

Region

Annotations can be placed either with respect to **Series** or **Chart**. by default, it will placed with respect to **Chart**.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ChartAnnotation, AnnotationsDirective, AnnotationDirective, Legend,
Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-react-
charts';
import { columnData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Category', title: 'Countries' };
    const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
    const border = { width: 2, color: 'grey' };
    const animation = { enable: true, duration: 1200, delay: 100 };
    const template = chartTemplate;
    function chartTemplate() {
        return (<div className='template'>
            <div>Highest Medal Count</div>
        </div>);
    }
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Olympic Medals'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
ChartAnnotation]}/>
        <AnnotationsDirective>
            <AnnotationDirective content={template} region='Series'
coordinateUnits='Point' x='Japan' y={75}>
                </AnnotationDirective>
            </AnnotationsDirective>
            <SeriesCollectionDirective>
                <SeriesDirective dataSource={columnData} xName='country'
yName='gold' name='Gold' border={border} animation={animation}
type='Column'>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
    }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, ChartAnnotation, AnnotationsDirective,
AnnotationDirective, Legend, Category, Tooltip, DataLabel, ColumnSeries}
from '@syncfusion/ej2-react-charts';
import { columnData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
    const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };

```

```

const border = { width: 2, color: 'grey' };
const animation = { enable: true, duration: 1200, delay: 100 };
const template: any = chartTemplate;
function chartTemplate(): any {
    return (<div className='template'>
        <div>Highest Medal Count</div>
    </div>);
}
return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
ChartAnnotation]} />
    <AnnotationsDirective>
        <AnnotationDirective content={template} region='Series'
coordinateUnits='Point' x='Japan' y={75}>
    </AnnotationDirective>
    </AnnotationsDirective>
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={columnData} xName='country'
yName='gold' name='Gold'
        border={border}
        animation={animation} type='Column'>
    </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Co-ordinate Units

Specified the coordinates units of the annotation either **Pixel** or **Point**.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ChartAnnotation, AnnotationsDirective, AnnotationDirective, Legend,
Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-react-
charts';
import { columnData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Category', title: 'Countries' };
    const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
    const border = { width: 2, color: 'grey' };
    const animation = { enable: true, duration: 1200, delay: 100 };
    const template = chartTemplate;
    function chartTemplate() {
        return (<div className='charttemplate'>
            <div> Annotation In Pixel </div>
        </div>);
    }
}

```

```

    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
ChartAnnotation]} />
    <AnnotationsDirective>
        <AnnotationDirective content={template} coordinateUnits='Pixel '
x={150} y={50}>
        </AnnotationDirective>
    </AnnotationsDirective>
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={columnData} xName='country'
yName='gold' name='Gold' border={border} animation={animation}
type='Column'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, ChartAnnotation, AnnotationsDirective,
AnnotationDirective, Legend, Category, Tooltip, DataLabel,
ColumnSeries } from '@syncfusion/ej2-react-charts';
import { columnData } from 'datasource.ts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
    const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
    const border = { width: 2, color: 'grey' };
    const animation = { enable: true, duration: 1200, delay: 100 };
    const template: any = chartTemplate;
    function chartTemplate(): any {
        return (<div className='charttemplate'>
            <div> Annotation In Pixel </div>
        </div>);
    }
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Olympic Medals'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
ChartAnnotation]} />
        <AnnotationsDirective>
            <AnnotationDirective content={template}
                coordinateUnits='Pixel ' x={150} y={50}>
            </AnnotationDirective>
        </AnnotationsDirective>
        <SeriesCollectionDirective>

```

```

        <SeriesDirective dataSource={columnData} xName='country'
yName='gold' name='Gold'
        border={border}
        animation={animation} type='Column'>
    </SeriesDirective>
</SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

- [Show total stacking values in data label](#)
- [Create footer and watermark for chart](#)[Link to the Video](#)

Legend in React Chart component

Legend provides information about the series rendered in the chart.

To get start quickly with Legends in React Charts, you can check on this video:

Position and Alignment

By using the [position](#) property, you can position the legend at left, right, top or bottom of the chart. The legend is positioned at the bottom of the chart, by default.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
function App() {
    const data = [
        { country: "USA", gold: 50, silver: 70, bronze: 45 },
        { country: "China", gold: 40, silver: 60, bronze: 55 },
        { country: "Japan", gold: 70, silver: 60, bronze: 50 },
        { country: "Australia", gold: 60, silver: 56, bronze: 40 },
        { country: "France", gold: 50, silver: 45, bronze: 35 },
        { country: "Germany", gold: 40, silver: 30, bronze: 22 },
        { country: "Italy", gold: 40, silver: 35, bronze: 37 },
        { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
    ];
    const primaryxAxis = { valueType: 'Category', title: 'Countries' };
    const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
    const legendSettings = { visible: true, position: 'Top' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} legendSettings={legendSettings} title='Olympic
Medals'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' type='Column'>
                </SeriesDirective>

```

```

        <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver' type='Column'>
        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze' type='Column'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel,
    Legend, Category, Tooltip, DataLabel, ColumnSeries}
from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { country: "USA", gold: 50, silver: 70, bronze: 45 },
        { country: "China", gold: 40, silver: 60, bronze: 55 },
        { country: "Japan", gold: 70, silver: 60, bronze: 50 },
        { country: "Australia", gold: 60, silver: 56, bronze: 40 },
        { country: "France", gold: 50, silver: 45, bronze: 35 },
        { country: "Germany", gold: 40, silver: 30, bronze: 22 },
        { country: "Italy", gold: 40, silver: 35, bronze: 37 },
        { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
    ];
    const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
    const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
    const legendSettings: LegendSettingsModel = { visible: true, position:
'Top' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        legendSettings={legendSettings}
        title='Olympic Medals'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver' type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>

```

```

    </ChartComponent>
  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));

```

Custom position helps you to position the legend anywhere in the chart using x, y coordinates.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
function App() {
  const data = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  const primaryxAxis = { valueType: 'Category', title: 'Countries' };
  const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
  const legendSettings = {
    visible: true, position: 'Custom',
    location: { x: 200, y: 40 }
  };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} legendSettings={legendSettings} title='Olympic
Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel,
Legend, Category, Tooltip, DataLabel, ColumnSeries }
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
  const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
  const legendSettings: LegendSettingsModel = {
    visible: true, position: 'Custom',
    location: { x: 200, y: 40 }
  };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    legendSettings={legendSettings}
    title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
  </return>
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));

```

Legend Reverse

You can reverse the order of the legend items by using the [reverse](#) property. By default, legend for the first series in the collection will be placed first.

INDEX.JSX

```

{% raw %}
import * as React from "react";

```



```

import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, ColumnSeries, DataLabel } from '@syncfusion/ej2-
react-charts';
function App() {
    const data = [
        { country: 'USA', gold: 50, silver: 70, bronze: 45 },
        { country: 'China', gold: 40, silver: 60, bronze: 55 },
        { country: 'Japan', gold: 70, silver: 60, bronze: 50 },
        { country: 'Australia', gold: 60, silver: 56, bronze: 40 },
        { country: 'France', gold: 50, silver: 45, bronze: 35 },
        { country: 'Germany', gold: 40, silver: 30, bronze: 22 },
        { country: 'Italy', gold: 40, silver: 35, bronze: 37 },
        { country: 'Sweden', gold: 30, silver: 25, bronze: 27 },
    ];
    return (<ChartComponent id="charts" primaryXAxis={{
        valueType: 'Category',
        interval: 1,
        majorGridLines: { width: 0 },
    }} primaryYAxis={{
        majorGridLines: { width: 0 },
        majorTickLines: { width: 0 },
        lineStyle: { width: 0 },
        labelStyle: { color: 'transparent' },
    }} title="Olympic Medals" legendSettings={{
        visible: true,
        reverse: true
    }}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName="country" yName="gold"
name="Gold" type="Column"></SeriesDirective>
            <SeriesDirective dataSource={data} xName="country" yName="silver"
name="Silver" type="Column"></SeriesDirective>
            <SeriesDirective dataSource={data} xName="country" yName="bronze"
name="Bronze" type="Column"></SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>);
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
    ChartComponent,
    SeriesCollectionDirective,
    SeriesDirective,
    Inject,
    Legend,

```

```

    Category,
    Tooltip,
    ColumnSeries,
    DataLabel }
from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { country: 'USA', gold: 50, silver: 70, bronze: 45 },
        { country: 'China', gold: 40, silver: 60, bronze: 55 },
        { country: 'Japan', gold: 70, silver: 60, bronze: 50 },
        { country: 'Australia', gold: 60, silver: 56, bronze: 40 },
        { country: 'France', gold: 50, silver: 45, bronze: 35 },
        { country: 'Germany', gold: 40, silver: 30, bronze: 22 },
        { country: 'Italy', gold: 40, silver: 35, bronze: 37 },
        { country: 'Sweden', gold: 30, silver: 25, bronze: 27 },
    ];

    return (
        <ChartComponent
            id="charts"
            primaryXAxis={{
                valueType: 'Category',
                interval: 1,
                majorGridLines: { width: 0 },
            }}
            primaryYAxis={{
                majorGridLines: { width: 0 },
                majorTickLines: { width: 0 },
                lineStyle: { width: 0 },
                labelStyle: { color: 'transparent' },
            }}
            title="Olympic Medals"
            legendSettings={{
                visible: true,
                reverse: true
            }}
        >
            <Inject
                services={[ColumnSeries, Legend, Tooltip, DataLabel, Category]}
            />
            <SeriesCollectionDirective>
                <SeriesDirective
                    dataSource={data}
                    xName="country"
                    yName="gold"
                    name="Gold"
                    type="Column"
                ></SeriesDirective>
                <SeriesDirective
                    dataSource={data}
                    xName="country"
                    yName="silver"
                    name="Silver"
                    type="Column"
                ></SeriesDirective>
                <SeriesDirective
                    dataSource={data}

```

```

        xName="country"
        yName="bronze"
        name="Bronze"
        type="Column"
    ></SeriesDirective>
</SeriesCollectionDirective>
</ChartComponent>
);
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

<!-- markdownlint-disable MD036 -->

Legend Alignment

<!-- markdownlint-disable MD036 -->

You can align the legend as **center**, **far** or **near** to the chart using [alignment](#) property.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
function App() {
    const data = [
        { country: "USA", gold: 50, silver: 70, bronze: 45 },
        { country: "China", gold: 40, silver: 60, bronze: 55 },
        { country: "Japan", gold: 70, silver: 60, bronze: 50 },
        { country: "Australia", gold: 60, silver: 56, bronze: 40 },
        { country: "France", gold: 50, silver: 45, bronze: 35 },
        { country: "Germany", gold: 40, silver: 30, bronze: 22 },
        { country: "Italy", gold: 40, silver: 35, bronze: 37 },
        { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
    ];
    const primaryxAxis = { valueType: 'Category', title: 'Countries' };
    const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
    const legendSettings = { position: 'Top', alignment: 'Near' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} legendSettings={legendSettings} title='Olympic
Medals'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver' type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze' type='Column'>

```

```

        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
  SeriesDirective, Inject, LegendSettingsModel,
  Legend, Category, Tooltip, DataLabel, ColumnSeries }
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
  const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
  const legendSettings: LegendSettingsModel = { position: 'Top', alignment:
'Near' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    legendSettings={legendSettings}
    title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Customization

To change the legend icon shape, you can use [legendShape](#) property in the [series](#). By default legend icon shape is [seriesType](#).

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
function App() {
    const data = [
        { country: "USA", gold: 50, silver: 70, bronze: 45 },
        { country: "China", gold: 40, silver: 60, bronze: 55 },
        { country: "Japan", gold: 70, silver: 60, bronze: 50 },
        { country: "Australia", gold: 60, silver: 56, bronze: 40 },
        { country: "France", gold: 50, silver: 45, bronze: 35 },
        { country: "Germany", gold: 40, silver: 30, bronze: 22 },
        { country: "Italy", gold: 40, silver: 35, bronze: 37 },
        { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
    ];
    const primaryxAxis = { valueType: 'Category', title: 'Countries' };
    const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
    const legendSettings = { position: 'Top' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} legendSettings={legendSettings} title='Olympic
Medals'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' legendShape='Circle' type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver' legendShape='SeriesType' type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze' legendShape='Rectangle' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel,
Legend, Category, Tooltip, DataLabel, ColumnSeries}
from '@syncfusion/ej2-react-charts';
```

```
function App() {
  const data: any[] = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category', title: 'Countries' };
  const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20, title: 'Medals' };
  const legendSettings: LegendSettingsModel = { position: 'Top' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    legendSettings={legendSettings}
    title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold'
        legendShape='Circle' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver'
        legendShape='SeriesType' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze'
        legendShape='Rectangle' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
</return>
  <export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));
```

Legend Size

By default, legend takes 20% - 25% of the chart's height horizontally, when it is placed on top or bottom position and 20% - 25% of the chart's width vertically, when placed on left or right position of the chart. You can change this default legend size by using the [width](#) and [height](#) property of the `legendSettings`.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject, Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-react-charts';
function App() {
  const data = [
```

```

    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  const primaryxAxis = { valueType: 'Category', title: 'Countries' };
  const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
  const legendSettings = { visible: true, height: '100', width: '500' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} legendSettings={legendSettings} title='Olympic
Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' legendShape='Circle' type='Column'>
    </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver' legendShape='SeriesType' type='Column'>
    </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze' legendShape='Rectangle' type='Column'>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel,
Legend, Category, Tooltip, DataLabel, ColumnSeries}
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };

```

```

const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
const legendSettings: LegendSettingsModel = { visible: true, height:
'100', width: '500' };
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  legendSettings={legendSettings}
  title='Olympic Medals'>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold'
      legendShape='Circle' type='Column'>
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver'
      legendShape='SeriesType' type='Column'>
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze'
      legendShape='Rectangle' type='Column'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Legend Item Size

You can customize the size of the legend items by using the [shapeHeight](#) and [shapeWidth](#) property.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
function App() {
  const data = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  const primaryxAxis = { valueType: 'Category', title: 'Countries' };
  const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
  const legendSettings = { visible: true, shapeHeight: 12, shapeWidth: 12
};

```



```

    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} legendSettings={legendSettings} title='Olympic
Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' legendShape='Circle' type='Column'>
        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver' legendShape='SeriesType' type='Column'>
        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze' legendShape='Rectangle' type='Column'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel,
Legend, Category, Tooltip, DataLabel, ColumnSeries}
from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { country: "USA", gold: 50, silver: 70, bronze: 45 },
        { country: "China", gold: 40, silver: 60, bronze: 55 },
        { country: "Japan", gold: 70, silver: 60, bronze: 50 },
        { country: "Australia", gold: 60, silver: 56, bronze: 40 },
        { country: "France", gold: 50, silver: 45, bronze: 35 },
        { country: "Germany", gold: 40, silver: 30, bronze: 22 },
        { country: "Italy", gold: 40, silver: 35, bronze: 37 },
        { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
    ];
    const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
    const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
    const legendSettings: LegendSettingsModel = { visible: true, shapeHeight:
12, shapeWidth: 12 };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        legendSettings={legendSettings}
        title='Olympic Medals'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>

```

```

    <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold'
    legendShape='Circle' type='Column'>
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver'
    legendShape='SeriesType' type='Column'>
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze'
    legendShape='Rectangle' type='Column'>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Paging for Legend

Paging will be enabled by default, when the legend items exceeds the legend bounds. You can view each legend items by navigating between the pages using navigation buttons.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
function App() {
    const data = [
        { x: 'WW', y: 12, y1: 22, y2: 38.3, y3: 50 },
        { x: 'EU', y: 9.9, y1: 26, y2: 45.2, y3: 63.6 },
        { x: 'APAC', y: 4.4, y1: 9.3, y2: 18.2, y3: 20.9 },
        { x: 'LATAM', y: 6.4, y1: 28, y2: 46.7, y3: 65.1 },
        { x: 'MEA', y: 30, y1: 45.7, y2: 61.5, y3: 73 },
        { x: 'NA', y: 25.3, y1: 35.9, y2: 64, y3: 81.4 }
    ];
    const primaryxAxis = {
        title: 'Countries', valueType: 'Category', interval: 1,
        labelIntersectAction: 'Rotate45'
    };
    const primaryyAxis = {
        title: 'Penetration (%)', rangePadding: 'None', labelFormat:
'{{value}}%',
        minimum: 0, maximum: 90
    };
    const legendSettings = {
        padding: 10, shapePadding: 10,
        visible: true, border: {
            width: 2, color: 'grey'
        },
        width: '200'
    };
    const marker1 = { visible: true, width: 10, height: 10, shape: 'Diamond'
};

```

```

    const marker2 = { visible: true, width: 10, height: 10, shape:
    'Pentagon' };
    const marker3 = { visible: true, width: 10, height: 10, shape:
    'Triangle' };
    const marker4 = { visible: true, width: 10, height: 10, shape: 'Circle'
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis} legendSettings={legendSettings} title='Olympic
    Medals'>
      <Inject services={[LineSeries, Legend, Tooltip, DataLabel,
    Category]}/>
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y' type='Line'
    width={2} name='December 2007' marker={marker1}>
        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y1' type='Line'
    width={2} name='December 2008' marker={marker2}>
        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y2' type='Line'
    width={2} name='December 2009' marker={marker3}>
        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y3' type='Line'
    width={2} name='December 2010' marker={marker4}>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
    SeriesDirective, Inject, LegendSettingsModel,
    Legend, Category, Tooltip, DataLabel, LineSeries }
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { x: 'WW', y: 12, y1: 22, y2: 38.3, y3: 50 },
    { x: 'EU', y: 9.9, y1: 26, y2: 45.2, y3: 63.6 },
    { x: 'APAC', y: 4.4, y1: 9.3, y2: 18.2, y3: 20.9 },
    { x: 'LATAM', y: 6.4, y1: 28, y2: 46.7, y3: 65.1 },
    { x: 'MEA', y: 30, y1: 45.7, y2: 61.5, y3: 73 },
    { x: 'NA', y: 25.3, y1: 35.9, y2: 64, y3: 81.4 }
  ];
  const primaryxAxis: AxisModel = {
    title: 'Countries', valueType: 'Category', interval: 1,
    labelIntersectAction: 'Rotate45'
  };
  const primaryyAxis: AxisModel = {
    title: 'Penetration (%)', rangePadding: 'None', labelFormat: '{value}%',
    minimum: 0, maximum: 90
  };

```

```

const legendSettings: LegendSettingsModel = {
  padding: 10, shapePadding: 10,
  visible: true, border: {
    width: 2, color: 'grey'
  },
  width: '200'
};
const marker1 = { visible: true, width: 10, height: 10, shape: 'Diamond'
};
const marker2 = { visible: true, width: 10, height: 10, shape: 'Pentagon'
};
const marker3 = { visible: true, width: 10, height: 10, shape: 'Triangle'
};
const marker4 = { visible: true, width: 10, height: 10, shape: 'Circle' };
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  legendSettings={legendSettings}
  title='Olympic Medals'>
  <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category]}
/>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y' type='Line'
width={2}
      name='December 2007' marker={marker1} >
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y1' type='Line'
width={2}
      name='December 2008' marker={marker2} >
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y2' type='Line'
width={2}
      name='December 2009' marker={marker3} >
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y3' type='Line'
width={2}
      name='December 2010' marker={marker4} >
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Legend Text Wrap

When the legend text exceeds the container, the text can be wrapped by using [textWrap](#) Property. End user can also wrap the legend text based on the [maximumLabelWidth](#) property.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
function App() {

```

```

const data = [
  { country: "USA", gold: 50, silver: 70, bronze: 45 },
  { country: "China", gold: 40, silver: 60, bronze: 55 },
  { country: "Japan", gold: 70, silver: 60, bronze: 50 },
  { country: "Australia", gold: 60, silver: 56, bronze: 40 },
  { country: "France", gold: 50, silver: 45, bronze: 35 },
  { country: "Germany", gold: 40, silver: 30, bronze: 22 },
  { country: "Italy", gold: 40, silver: 35, bronze: 37 },
  { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
];
const primaryxAxis = { valueType: 'Category', title: 'Countries' };
const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
const legendSettings = { visible: true, position: 'Right', textWrap:
'Wrap', maximumLabelWidth: 50, };
return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} legendSettings={legendSettings} title='Olympic
Medals'>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold Medals' type='Column'>
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver Medals' type='Column'>
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze Medals' type='Column'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel,
Legend, Category, Tooltip, DataLabel, ColumnSeries}
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];

```

```

const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
const legendSettings: LegendSettingsModel = { visible: true, position:
'Right', textWrap: 'Wrap', maximumLabelWidth: 50, };
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  legendSettings={legendSettings}
  title='Olympic Medals'>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold Medals' type='Column'>
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver Medals' type='Column'>
    </SeriesDirective>
    <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze Medals' type='Column'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Legend text color

The text color of the legend can be changed by setting the [color](#) property to the desired color in the [textStyle](#) property of the legendSettings.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
class App extends React.Component {
  data = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  primaryxAxis = { valueType: 'Category', title: 'Countries' };
  primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title: 'Medals'
};
  legendSettings = { visible: true, textStyle: { color: "red" } };
  render() {

```

```

        return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} legendSettings={legendSettings} title='Olympic
Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' type='Column'>
        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver' type='Column'>
        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze' type='Column'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
    }
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel,
Legend, Category, Tooltip, DataLabel, ColumnSeries}
from '@syncfusion/ej2-react-charts';
class App extends React.Component<{}, {}> {
    public data: any[] = [
        { country: "USA", gold: 50, silver: 70, bronze: 45 },
        { country: "China", gold: 40, silver: 60, bronze: 55 },
        { country: "Japan", gold: 70, silver: 60, bronze: 50 },
        { country: "Australia", gold: 60, silver: 56, bronze: 40 },
        { country: "France", gold: 50, silver: 45, bronze: 35 },
        { country: "Germany", gold: 40, silver: 30, bronze: 22 },
        { country: "Italy", gold: 40, silver: 35, bronze: 37 },
        { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
    ];
    public primaryxAxis: AxisModel = {
        valueType: 'Category', title: 'Countries'
    };
    public primaryyAxis: AxisModel = {
        minimum: 0, maximum: 80, interval: 20, title: 'Medals'
    };
    public legendSettings: LegendSettingsModel = {
        visible: true, textStyle: { color: "red" }
    };
    render() {
        return <ChartComponent id='charts' primaryXAxis={this.primaryxAxis}
primaryYAxis={this.primaryyAxis} legendSettings={legendSettings}
title='Olympic Medals'>

```

```

    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={this.data} xName='country' yName='gold'
name='Gold' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={this.data} xName='country' yName='silver'
name='Silver' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={this.data} xName='country' yName='bronze'
name='Bronze' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
}
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Set the label color based on series color

You can set the legend label color based on series color by using chart's [loaded](#) event.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
// declare the series colors
let colors = ['#00BDAE', '#404041', '#357CD2'];
function App() {
  const data = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  const primaryxAxis = { valueType: 'Category', title: 'Countries' };
  const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
  const legendSettings = { visible: true, position: 'Top' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} legendSettings={legendSettings}
loaded={onChartLoaded.bind(this)} title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' type='Column'>
      </SeriesDirective>

```



```

        <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver' type='Column'>
        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze' type='Column'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
function onChartLoaded(args) {
    let chart = document.getElementById('charts');
    let legendTextCol =
chart.querySelectorAll('[id*="chart_legend_text_"]');
    for (let i = 0; i < legendTextCol.length; i++) {
        //set the color to legend label
        legendTextCol[i].setAttribute('fill', colors[i]);
    }
}
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, Chart, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel,
    Legend, Category, Tooltip, DataLabel, ColumnSeries,
ILoadedEventArgs }
from '@syncfusion/ej2-react-charts';
// declare the series colors
let colors: string[] = ['#00BDAE', '#404041', '#357CD2'];
function App() {
    const data: any[] = [
        { country: "USA", gold: 50, silver: 70, bronze: 45 },
        { country: "China", gold: 40, silver: 60, bronze: 55 },
        { country: "Japan", gold: 70, silver: 60, bronze: 50 },
        { country: "Australia", gold: 60, silver: 56, bronze: 40 },
        { country: "France", gold: 50, silver: 45, bronze: 35 },
        { country: "Germany", gold: 40, silver: 30, bronze: 22 },
        { country: "Italy", gold: 40, silver: 35, bronze: 37 },
        { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
    ];
    const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
    const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
    const legendSettings: LegendSettingsModel = { visible: true, position:
'Top' };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        legendSettings={legendSettings}
        loaded={onChartLoaded.bind(this)}
        title='Olympic Medals'>

```

```

    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
  function onChartLoaded(args: ILoadedEventArgs) {
    let chart: Chart = document.getElementById('charts');
    let legendTextCol: HTMLElement =
chart.querySelectorAll('[id*="chart_legend_text_"]');
    for (let i = 0; i < legendTextCol.length; i++) {
      //set the color to legend label
      legendTextCol[i].setAttribute('fill', colors[i]);
    }
  }
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Series Selection on Legend

By default, legend click enables you to collapse the series visibility. On other hand, if you need to select a series through legend click, disable the [toggleVisibility](#).

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries, Selection } from
'@syncfusion/ej2-react-charts';
function App() {
  const data = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  const primaryxAxis = { valueType: 'Category', title: 'Countries' };
  const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
  const legendSettings = { visible: true, toggleVisibility: false };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} legendSettings={legendSettings}
selectionMode='Series' title='Olympic Medals'>

```

```

    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
Selection]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' legendShape='Circle' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver' legendShape='SeriesType' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze' legendShape='Rectangle' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel,
Legend, Category, Tooltip, DataLabel, ColumnSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
  const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
  const legendSettings: LegendSettingsModel = { visible: true,
toggleVisibility: false };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    legendSettings={legendSettings}
    selectionMode='Series'
    title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
Selection]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold'
        legendShape='Circle' type='Column'>

```

```

        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='country' yName='silver'
name='Silver'
        legendShape='SeriesType' type='Column'>
        </SeriesDirective>
        <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze'
        legendShape='Rectangle' type='Column'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Enable Animation

You can customize the animation while clicking legend by setting `enableAnimation` as true or false using `enableAnimation` property in chart.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries, Selection } from
'@syncfusion/ej2-react-charts';
function App() {
    const chartData1 = [{ x: 'USA', y: 46 }, { x: 'GBR', y: 27 }, { x:
'CHN', y: 26 }];
    const chartData2 = [{ x: 'USA', y: 37 }, { x: 'GBR', y: 23 }, { x:
'CHN', y: 18 }];
    const chartData3 = [{ x: 'USA', y: 38 }, { x: 'GBR', y: 17 }, { x:
'CHN', y: 26 }];
    const primaryxAxis = { valueType: 'Category', interval: 1,
majorGridLines: { width: 0 } };
    const primaryyAxis = { majorGridLines: { width: 0 },
        majorTickLines: { width: 0 }, lineStyle: { width: 0 }, labelStyle: {
color: 'transparent' } };
    const marker = { dataLabel: { visible: true, position: 'Top', font: {
fontWeight: '600', color: '#ffffff' } } };
    const tooltip = { enable: true };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Olympic Medal Counts - RIO'
enableAnimation={true} tooltip={tooltip}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
Selection]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData1} xName='x' yName='y'
name='Gold' marker={marker} type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={chartData2} xName='x' yName='y'
name='Silver' marker={marker} type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={chartData3} xName='x' yName='y'
name='Bronze' marker={marker} type='Column'>
            </SeriesDirective>

```

```

        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel, TooltipSettingsModel,
Legend, Category, Tooltip, DataLabel, ColumnSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
    const chartData1: any[] = [{ x: 'USA', y: 46 }, { x: 'GBR', y: 27 }, { x:
'CHN', y: 26 }];
    const chartData2: any[] = [{ x: 'USA', y: 37 }, { x: 'GBR', y: 23 }, { x:
'CHN', y: 18 }];
    const chartData3: any[] = [{ x: 'USA', y: 38 }, { x: 'GBR', y: 17 }, { x:
'CHN', y: 26 }];
    const primaryXAxis: AxisModel = {valueType: 'Category', interval: 1,
majorGridLines: { width: 0 } };
    const primaryYAxis: AxisModel = {majorGridLines: { width: 0 },
majorTickLines: { width: 0 }, lineStyle: { width: 0 },
labelStyle: { color: 'transparent' }};
    const marker = { dataLabel: { visible: true, position: 'Top', font: {
fontWeight: '600', color: '#ffffff' } } };
    const tooltip: TooltipSettingsModel = { enable: true };
    return <ChartComponent id='charts'
        primaryXAxis={primaryXAxis}
        primaryYAxis={primaryYAxis}
        title='Olympic Medal Counts - RIO'
        enableAnimation={true}
        tooltip={tooltip}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
Selection]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData1} xName='x' yName='y'
name='Gold'
                marker={marker} type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={chartData2} xName='x' yName='y'
name='Silver'
                marker={marker} type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={chartData3} xName='x' yName='y'
name='Bronze'
                marker={marker} type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;

```

```
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

Collapsing Legend Item

By default, series name will be displayed as legend. To skip the legend for a particular series, you can give empty string to the series name.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
function App() {
  const data = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  const primaryxAxis = { valueType: 'Category', title: 'Countries' };
  const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
  const legendSettings = { visible: true, toggleVisibility: true };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} legendSettings={legendSettings} title='Olympic
Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' legendShape='Circle' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='silver'
type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze' legendShape='Rectangle' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
```

```

import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel,
      Legend, Category, Tooltip, DataLabel, ColumnSeries}
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
  const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
  const legendSettings: LegendSettingsModel = { visible: true,
toggleVisibility: true };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    legendSettings={legendSettings}
    title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold'
        legendShape='Circle' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='silver'
type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze'
        legendShape='Rectangle' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Legend Title

You can set title for legend using `title` property in `legendSettings`. You can also customize the `fontStyle`, `size`, `fontWeight`, `color`, `textAlignment`, `fontFamily`, `opacity` and `textOverflow` of legend title. `titlePosition` is used to set the legend position in `Top`, `Left` and `Right` position. `maximumTitleWidth` is used to set the width of the legend title. By default, it will be `100px`.

INDEX.JSX

```
import * as React from "react";
```

```

import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
function App() {
    const data = [
        { country: "USA", gold: 50, silver: 70, bronze: 45 },
        { country: "China", gold: 40, silver: 60, bronze: 55 },
        { country: "Japan", gold: 70, silver: 60, bronze: 50 },
        { country: "Australia", gold: 60, silver: 56, bronze: 40 },
        { country: "France", gold: 50, silver: 45, bronze: 35 },
        { country: "Germany", gold: 40, silver: 30, bronze: 22 },
        { country: "Italy", gold: 40, silver: 35, bronze: 37 },
        { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
    ];
    const primaryXAxis = {
        valueType: 'Category'
    };
    const primaryYAxis = {
        minimum: 0, maximum: 80, interval: 20, title: 'Medals'
    };
    const legendSettings = {
        visible: true, title: 'Countries'
    };
    return <ChartComponent id='charts' primaryXAxis={primaryXAxis}
primaryYAxis={primaryYAxis} legendSettings={legendSettings} title='Olympic
Medals'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='country' yName='gold'
type='Column' name='December 2007'>
            </SeriesDirective>
            <SeriesDirective dataSource={data} xName='country' yName='silver'
type='Column' name='December 2008'>
            </SeriesDirective>
            <SeriesDirective dataSource={data} xName='country' yName='bronze'
type='Column' name='December 2009'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel,
Legend, Category, Tooltip, DataLabel, ColumnSeries}
from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [

```



```

    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  const primaryxAxis: AxisModel = {
    valueType: 'Category'
  };
  const primaryyAxis: AxisModel = {
    minimum: 0, maximum: 80, interval: 20, title: 'Medals'
  };
  const legendSettings: LegendSettingsModel = {
    visible: true, title: 'Countries'
  };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    legendSettings={legendSettings}
    title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
type='Column'
        name='December 2007'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='silver'
type='Column'
        name='December 2008'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='bronze'
type='Column'
        name='December 2009'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
</return>
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Arrow Page Navigation

By default, the page number will be enabled while legend paging. Now, you can disable that page number and also you can get left and right arrows for page navigation. You have to set `false` value to `enablePages` to get this support.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';

```

```
function App() {
  const data = [
    { x: 'WW', y: 12, y1: 22, y2: 38.3, y3: 50 },
    { x: 'EU', y: 9.9, y1: 26, y2: 45.2, y3: 63.6 },
    { x: 'APAC', y: 4.4, y1: 9.3, y2: 18.2, y3: 20.9 },
    { x: 'LATAM', y: 6.4, y1: 28, y2: 46.7, y3: 65.1 },
    { x: 'MEA', y: 30, y1: 45.7, y2: 61.5, y3: 73 },
    { x: 'NA', y: 25.3, y1: 35.9, y2: 64, y3: 81.4 }
  ];
  const primaryxAxis = {
    title: 'Countries', valueType: 'Category', interval: 1,
    labelIntersectAction: 'Rotate45'
  };
  const primaryyAxis = {
    title: 'Penetration (%)', rangePadding: 'None', labelFormat:
    '{value}%',
    minimum: 0, maximum: 90
  };
  const legendSettings = {
    width: '180', enablePages: false
  };
  const marker1 = { visible: true, width: 10, height: 10, shape: 'Diamond' };
  const marker2 = { visible: true, width: 10, height: 10, shape:
  'Pentagon' };
  const marker3 = { visible: true, width: 10, height: 10, shape:
  'Triangle' };
  const marker4 = { visible: true, width: 10, height: 10, shape: 'Circle' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis} legendSettings={legendSettings} title='Olympic
  Medals'>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel,
  Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' type='Line'
  width={2} name='December 2007' marker={marker1}>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y1' type='Line'
  width={2} name='December 2008' marker={marker2}>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y2' type='Line'
  width={2} name='December 2009' marker={marker3}>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y3' type='Line'
  width={2} name='December 2010' marker={marker4}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel,
Legend, Category, Tooltip, DataLabel, LineSeries }
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { x: 'WW', y: 12, y1: 22, y2: 38.3, y3: 50 },
    { x: 'EU', y: 9.9, y1: 26, y2: 45.2, y3: 63.6 },
    { x: 'APAC', y: 4.4, y1: 9.3, y2: 18.2, y3: 20.9 },
    { x: 'LATAM', y: 6.4, y1: 28, y2: 46.7, y3: 65.1 },
    { x: 'MEA', y: 30, y1: 45.7, y2: 61.5, y3: 73 },
    { x: 'NA', y: 25.3, y1: 35.9, y2: 64, y3: 81.4 }
  ];
  const primaryxAxis: AxisModel = {
    title: 'Countries', valueType: 'Category', interval: 1,
    labelIntersectAction: 'Rotate45'
  };
  const primaryyAxis: AxisModel = {
    title: 'Penetration (%)', rangePadding: 'None', labelFormat: '{value}%',
    minimum: 0, maximum: 90
  };
  const legendSettings: LegendSettingsModel = {
    width: '180', enablePages: false
  };
  const marker1 = { visible: true, width: 10, height: 10, shape: 'Diamond' };
  const marker2 = { visible: true, width: 10, height: 10, shape: 'Pentagon' };
  const marker3 = { visible: true, width: 10, height: 10, shape: 'Triangle' };
  const marker4 = { visible: true, width: 10, height: 10, shape: 'Circle' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    legendSettings={legendSettings}
    title='Olympic Medals'>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category]}
  />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' type='Line'
width={2}
        name='December 2007' marker={marker1} >
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y1' type='Line'
width={2}
        name='December 2008' marker={marker2} >
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y2' type='Line'
width={2}
        name='December 2009' marker={marker3} >
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y3' type='Line'
width={2}
        name='December 2010' marker={marker4} >
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>

```

```

    </SeriesCollectionDirective>
  </ChartComponent>

  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));

```

Legend Item Padding

The [itemPadding](#) property can be used to adjust the space between the legend items.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
class App extends React.Component {
  data = [
    { x: 'WW', y: 12, y1: 22, y2: 38.3, y3: 50 },
    { x: 'EU', y: 9.9, y1: 26, y2: 45.2, y3: 63.6 },
    { x: 'APAC', y: 4.4, y1: 9.3, y2: 18.2, y3: 20.9 },
    { x: 'LATAM', y: 6.4, y1: 28, y2: 46.7, y3: 65.1 },
    { x: 'MEA', y: 30, y1: 45.7, y2: 61.5, y3: 73 },
    { x: 'NA', y: 25.3, y1: 35.9, y2: 64, y3: 81.4 }
  ];
  primaryxAxis = {
    title: 'Countries', valueType: 'Category', interval: 1,
    labelIntersectAction: 'Rotate45'
  };
  primaryyAxis = {
    title: 'Penetration (%)', rangePadding: 'None', labelFormat:
    '{value}%',
    minimum: 0, maximum: 90
  };
  legendSettings = {
    width: '180', enablePages: false
  };
  marker1 = { visible: true, width: 10, height: 10, position: "Bottom",
shape: 'Diamond', itemPadding: 30 };
  marker2 = { visible: true, width: 10, height: 10, position: "Bottom",
shape: 'Pentagon', itemPadding: 30 };
  marker3 = { visible: true, width: 10, height: 10, position: "Bottom",
shape: 'Triangle', itemPadding: 30 };
  marker4 = { visible: true, width: 10, height: 10, position: "Bottom",
shape: 'Circle', itemPadding: 30 };
  render() {
    return <ChartComponent id='charts' primaryXAxis={this.primaryxAxis}
primaryYAxis={this.primaryyAxis} legendSettings={this.legendSettings}
title='Olympic Medals'>
      <Inject services={[LineSeries, Legend, Tooltip, DataLabel,
Category]} />
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={this.data} xName='x' yName='y'
type='Line' width={2} name='December 2007' marker={this.marker1}>
        </SeriesDirective>

```

```

        <SeriesDirective dataSource={this.data} xName='x' yName='y1'
type='Line' width={2} name='December 2008' marker={this.marker2}>
        </SeriesDirective>
        <SeriesDirective dataSource={this.data} xName='x' yName='y2'
type='Line' width={2} name='December 2009' marker={this.marker3}>
        </SeriesDirective>
        <SeriesDirective dataSource={this.data} xName='x' yName='y3'
type='Line' width={2} name='December 2010' marker={this.marker4}>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
    }
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel,
    Legend, Category, Tooltip, DataLabel, LineSeries }
from '@syncfusion/ej2-react-charts';
class App extends React.Component<{}, {}> {
    public data: any[] = [
        { x: 'WW', y: 12, y1: 22, y2: 38.3, y3: 50 },
        { x: 'EU', y: 9.9, y1: 26, y2: 45.2, y3: 63.6 },
        { x: 'APAC', y: 4.4, y1: 9.3, y2: 18.2, y3: 20.9 },
        { x: 'LATAM', y: 6.4, y1: 28, y2: 46.7, y3: 65.1 },
        { x: 'MEA', y: 30, y1: 45.7, y2: 61.5, y3: 73 },
        { x: 'NA', y: 25.3, y1: 35.9, y2: 64, y3: 81.4 }
    ];
    public primaryxAxis: AxisModel = {
        title: 'Countries', valueType: 'Category', interval: 1,
        labelIntersectAction: 'Rotate45'
    };
    public primaryyAxis: AxisModel = {
        title: 'Penetration (%)', rangePadding: 'None', labelFormat: '{value}%',
        minimum: 0, maximum: 90
    };
    public legendSettings: LegendSettingsModel = {
        width: '180', enablePages: false
    };
    public marker1 = { visible: true, width: 10, height: 10, position:
"Bottom", shape: 'Diamond', itemPadding: 30 };
    public marker2 = { visible: true, width: 10, height: 10, position:
"Bottom", shape: 'Pentagon', itemPadding: 30 };
    public marker3 = { visible: true, width: 10, height: 10, position:
"Bottom", shape: 'Triangle', itemPadding: 30 };
    public marker4 = { visible: true, width: 10, height: 10, position:
"Bottom", shape: 'Circle', itemPadding: 30 };
    render() {
        return <ChartComponent id='charts'
            primaryXAxis={this.primaryxAxis}

```

```

    primaryYAxis={this.primaryYAxis}
    legendSettings={this.legendSettings}
    title='Olympic Medals'>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category]}
  />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={this.data} xName='x' yName='y'
type='Line' width={2}
      name='December 2007' marker={this.marker1} >
    </SeriesDirective>
      <SeriesDirective dataSource={this.data} xName='x' yName='y1'
type='Line' width={2}
      name='December 2008' marker={this.marker2} >
    </SeriesDirective>
      <SeriesDirective dataSource={this.data} xName='x' yName='y2'
type='Line' width={2}
      name='December 2009' marker={this.marker3} >
    </SeriesDirective>
      <SeriesDirective dataSource={this.data} xName='x' yName='y3'
type='Line' width={2}
      name='December 2010' marker={this.marker4} >
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
}
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Note: To use legend feature, we need to inject **Legend** module into the **services**.

See Also

- [Customize each shape in legend](#)

Tooltip in React Chart component

Chart will display details about the points through tooltip, when the mouse is moved over the point

<!-- markdownlint-disable MD036 -->

Default tooltip

<!-- markdownlint-disable MD012 -->

By default, tooltip is not visible. You can enable the tooltip by setting [enable](#) property to **true** and by injecting **Tooltip** module into the **services**.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, StepLineSeries } from
'@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {

```

```

const primaryxAxis = { valueType: 'DateTime' };
const tooltip = { enable: true };
const marker = { visible: true, width: 10, height: 10 };
return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
tooltip={tooltip} title='Unemployment Rates 1975-2010'>
    <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
DateTime]}/>
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y' name='China'
width={2} type='StepLine' marker={marker}>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, TooltipSettingsModel,
Legend, DateTime, Tooltip, DataLabel, StepLineSeries}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime' };
    const tooltip: TooltipSettingsModel = { enable: true };
    const marker = { visible: true, width: 10, height: 10 };
    const data: any[] = [
        { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
        { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
        { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
        { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
        { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
        { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
        { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
        { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        tooltip={tooltip}
        title='Unemployment Rates 1975-2010'>
        <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
DateTime]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' name='China'
width={2}
                type='StepLine' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;

```

```
ReactDOM.render(<App />, document.getElementById("charts"));
```

Fixed tooltip

By default, tooltip track the mouse movement, but you can set a fixed position for the tooltip by using the [location](#) property.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
DateTime, Tooltip, StepLineSeries } from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis = { valueType: 'DateTime' };
    const tooltip = { enable: true, location: { x: 120, y: 20 } };
    const marker = { visible: true, width: 10, height: 10 };
    const data = [
        { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
        { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
        { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
        { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
        { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
        { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
        { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
        { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
    ];
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
        tooltip={tooltip} title='Unemployment Rates 1975-2010'>
        <Inject services={[StepLineSeries, Tooltip, DateTime]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' name='China'
                width={2} type='StepLine' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
    AxisModel, ChartComponent, SeriesCollectionDirective, SeriesDirective,
    Inject, TooltipSettingsModel,
    DateTime, Tooltip, StepLineSeries
}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime' };
    const tooltip: TooltipSettingsModel = { enable: true, location: { x: 120,
        y: 20 } };
    const marker = { visible: true, width: 10, height: 10 };
```



```

const data: any[] = [
  { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
  { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
  { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
  { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
  { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
  { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
  { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
  { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
];
return <ChartComponent id='charts'
  primaryXAxis={primaryXAxis}
  tooltip={tooltip}
  title='Unemployment Rates 1975-2010'>
  <Inject services={[StepLineSeries, Tooltip, DateTime]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y' name='China'
width={2}>
      type='StepLine' marker={marker}>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Format the tooltip

By default, tooltip shows information of x and y value in points. In addition to that, you can show more information in tooltip. For example the format `${series.name} ${point.x}` shows series name and point x value.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, StepLineSeries } from
'@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
  const primaryXAxis = { valueType: 'DateTime' };
  const tooltip = {
    enable: true, header: 'Unemployment',
    format: '<b>${point.x} : ${point.y}</b>'
  };
  const marker = { visible: true, width: 10, height: 10 };
  return <ChartComponent id='charts' primaryXAxis={primaryXAxis}
  tooltip={tooltip} title='Unemployment Rates 1975-2010'>
    <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' name='China'
width={2} type='StepLine' marker={marker}>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;

```

```

}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, TooltipSettingsModel,
Legend, DateTime, Tooltip, DataLabel, StepLineSeries }
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryXAxis: AxisModel = { valueType: 'DateTime' };
  const tooltip: TooltipSettingsModel = {
    enable: true, header: 'Unemployment',
    format: '<b>${point.x} : ${point.y}</b>'
  };
  const marker = { visible: true, width: 10, height: 10 };
  const data: any[] = [
    { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
    { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
    { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
    { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
    { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
    { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
    { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
    { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryXAxis}
    tooltip={tooltip}
    title='Unemployment Rates 1975-2010'>
    <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' name='China'
width={2}
        type='StepLine' marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

<!-- markdownlint-disable MD013 -->

Individual series format

You can format the each series tooltip separately using series [tooltipFormat](#) property.

Note: If series [tooltipFormat](#) is given, it shows the tooltip for that series in that format, or else it will take tooltip format.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, SplineSeries } from '@syncfusion/ej2-
react-charts';
function App() {
    const primaryxAxis = { valueType: 'Category' };
    const tooltip = {
        enable: true, header: 'Unemployment',
        format: '<b>${point.x} : ${point.y}</b>'
    };
    const marker = { visible: true, width: 10, height: 10 };
    const data1 = [
        { x: 'Sun', y: 15 }, { x: 'Mon', y: 22 },
        { x: 'Tue', y: 32 },
        { x: 'Wed', y: 31 },
        { x: 'Thu', y: 29 }, { x: 'Fri', y: 24 },
        { x: 'Sat', y: 18 },
    ];
    const data2 = [
        { x: 'Sun', y: 10 }, { x: 'Mon', y: 18 },
        { x: 'Tue', y: 28 },
        { x: 'Wed', y: 28 },
        { x: 'Thu', y: 26 }, { x: 'Fri', y: 20 },
        { x: 'Sat', y: 15 }
    ];
    const data3 = [
        { x: 'Sun', y: 2 }, { x: 'Mon', y: 12 },
        { x: 'Tue', y: 22 },
        { x: 'Wed', y: 23 },
        { x: 'Thu', y: 19 }, { x: 'Fri', y: 13 },
        { x: 'Sat', y: 8 },
    ];
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
tooltip={tooltip} title='Unemployment Rates 1975-2010'>
        <Inject services={[SplineSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data1} xName='x' yName='y' name='Max
Temp' width={2} type='Spline' marker={marker} tooltipFormat='${point.x}'>
            </SeriesDirective>
            <SeriesDirective dataSource={data2} xName='x' yName='y' name='Avg
Temp' width={2} type='Spline' marker={marker} tooltipFormat='${point.y}'>
            </SeriesDirective>
            <SeriesDirective dataSource={data3} xName='x' yName='y' name='Min
Temp' width={2} type='Spline' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, TooltipSettingsModel, Legend, Category, Tooltip,
DataLabel, SplineSeries }
from '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
  const primaryXAxis: AxisModel = { valueType: 'Category' };
  const tooltip: TooltipSettingsModel = {
    enable: true, header: 'Unemployment',
    format: '<b>${point.x} : ${point.y}</b>'
  };
  const marker = { visible: true, width: 10, height: 10 };
  const data1: object[] = [
    { x: 'Sun', y: 15 }, { x: 'Mon', y: 22 },
    { x: 'Tue', y: 32 },
    { x: 'Wed', y: 31 },
    { x: 'Thu', y: 29 }, { x: 'Fri', y: 24 },
    { x: 'Sat', y: 18 },
  ];
  const data2: object[] = [
    { x: 'Sun', y: 10 }, { x: 'Mon', y: 18 },
    { x: 'Tue', y: 28 },
    { x: 'Wed', y: 28 },
    { x: 'Thu', y: 26 }, { x: 'Fri', y: 20 },
    { x: 'Sat', y: 15 }
  ];
  const data3: object[] = [
    { x: 'Sun', y: 2 }, { x: 'Mon', y: 12 },
    { x: 'Tue', y: 22 },
    { x: 'Wed', y: 23 },
    { x: 'Thu', y: 19 }, { x: 'Fri', y: 13 },
    { x: 'Sat', y: 8 },
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryXAxis}
    tooltip={tooltip}
    title='Unemployment Rates 1975-2010'>
    <Inject services={[SplineSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data1} xName='x' yName='y' name='Max
Temp' width={2}
        type='Spline' marker={marker} tooltipFormat='${point.x}'>
      </SeriesDirective>
      <SeriesDirective dataSource={data2} xName='x' yName='y' name='Avg
Temp' width={2}
        type='Spline' marker={marker} tooltipFormat='${point.y}'>
      </SeriesDirective>
      <SeriesDirective dataSource={data3} xName='x' yName='y' name='Min
Temp' width={2}
        type='Spline' marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>

```

```

    </ChartComponent>
  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));

```

<!-- markdownlint-disable MD013 -->

Tooltip template

Any HTML elements can be displayed in the tooltip by using the [template](#) property of the tooltip. You can use the $\{x\}$ and $\{y\}$ as place holders in the HTML element to display the x and y values of the corresponding data point.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, StepLineSeries } from
"@syncfusion/ej2-react-charts";
import { data } from "datasource.ts";
function App() {
  const primaryxAxis = { valueType: 'DateTime' };
  const template = chartTemplate;
  const tooltip = {
    enable: true,
    template: template
  };
  function chartTemplate(args) {
    return (<div id="templateWrap">
      <table style={{ width: '100%', margin: '5px', border: '1px solid
black', backgroundColor: '#00FFFF' }}>
        <tbody><tr><th colspan={2}>Unemployment</th></tr>
        <tr><td>{args.x}</td><td>:</td><td>{args.y}</td></tr>
        </tbody>
      </table>
    </div>);
  }
  const marker = { visible: true, width: 10, height: 10 };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
tooltip={tooltip} title='Unemployment Rates 1975-2010'>
    <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' name='China'
width={2} type='StepLine' marker={marker}>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, TooltipSettingsModel,
Legend, DateTime, Tooltip, DataLabel, StepLineSeries}
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime' };
  const template:any = chartTemplate;
  const tooltip: TooltipSettingsModel = {
    enable: true,
    template: template
  };
  function chartTemplate(args:any){
    return (<div id="templateWrap">
      <table style={{width:'100%',margin: '5px', border: '1px solid black'
,backgroundColor:'#00FFFF'}}>
        <tbody><tr><th colSpan={2}>Unemployment</th></tr>
        <tr><td>{args.x}</td><td>:</td><td>{args.y}</td></tr>
        </tbody>
      </table>
    </div>);
  }
  const marker = { visible: true, width: 10, height: 10 };
  const data: any[] = [
    { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
    { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
    { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
    { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
    { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
    { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
    { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
    { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    tooltip={tooltip}
    title='Unemployment Rates 1975-2010'>
    <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' name='China'
width={2}
        type='StepLine' marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

Tooltip mapping name

By default, tooltip shows information of x and y value in points. You can show more information from data source in tooltip by using the [tooltipMappingName](#) property of the tooltip. You can use the `${point.tooltip}` as place holders to display the specified tooltip content.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Category, Legend, DateTime, Tooltip, DataLabel, ColumnSeries } from
'@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const primaryXAxis = { valueType: 'Category' };
    const tooltip = { enable: true, format: '${point.tooltip}' };
    const legendSettings = { visible: false };
    return <ChartComponent id='charts' primaryXAxis={primaryXAxis}
tooltip={tooltip} legendSettings={legendSettings} title='Internet Users in
Million - 2016'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, DateTime,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} xName='x' yName='y'
type='Column' tooltipMappingName='country'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, TooltipSettingsModel, LegendSettingsModel, Category,
Legend, DateTime, Tooltip, DataLabel, ColumnSeries }
from '@syncfusion/ej2-react-charts';
import { chartData } from 'datasource.ts';
function App() {
    const primaryXAxis: AxisModel = { valueType: 'Category' };
    const tooltip: TooltipSettingsModel = { enable: true, format:
'${point.tooltip}' };
    const legendSettings: LegendSettingsModel = { visible: false };
    return <ChartComponent id='charts'
primaryXAxis={primaryXAxis}
tooltip={tooltip}
legendSettings={legendSettings}
title='Internet Users in Million - 2016'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, DateTime,
Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={chartData} xName='x' yName='y'
type='Column' tooltipMappingName='country'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

```

        type='Column' tooltipMappingName='country'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Customize the appearance of tooltip

The [fill](#) and [border](#) properties are used to customize the background color and border of the tooltip respectively. The [textStyle](#) property in the tooltip is used to customize the font of the tooltip text. The [highlightColor](#) property is used to customize the point color while hovering for tooltip.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, StepLineSeries } from
"@syncfusion/ej2-react-charts";
import { data } from "datasource.ts";
function App() {
  const primaryxAxis = { valueType: 'DateTime' };
  const tooltip = {
    enable: true, format: '${series.name} ${point.x} : ${point.y}',
    fill: '#7bb4eb', border: {
      width: 2,
      color: 'grey'
    }
  };
  const marker = { visible: true, width: 10, height: 10 };
  const titlestyle = {
    fontFamily: "Arial", fontStyle: 'italic', fontWeight: 'regular',
    color: "#E27F2D", size: '23px'
  };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
  tooltip={tooltip} title='Unemployment Rates 1975-2010' highlightColor='red'
  titleStyle={titlestyle}>
    <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
    DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' name='China'
      width={2} type='StepLine' marker={marker}>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";

```



```

import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, TooltipSettingsModel,
      Legend, DateTime, Tooltip, DataLabel, StepLineSeries }
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime' };
  const tooltip: TooltipSettingsModel = {
    enable: true, format: '${series.name} ${point.x} : ${point.y}',
    fill: '#7bb4eb', border: {
      width: 2,
      color: 'grey'
    }
  };
  const marker = { visible: true, width: 10, height: 10 };
  const titlestyle = {
    fontFamily: "Arial", fontStyle: 'italic', fontWeight: 'regular',
    color: "#E27F2D", size: '23px'
  };
  const data: any[] = [
    { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
    { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
    { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
    { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
    { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
    { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
    { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
    { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    tooltip={tooltip}
    title='Unemployment Rates 1975-2010'
    highlightColor='red'
    titleStyle={titlestyle}>
    <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' name='China'
width={2}
        type='StepLine' marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
  </return>
  export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

See also

- [Format the tooltip value](#)
- [Create a table in tooltipLink to the Video](#)

Zooming in React Chart component

To get start quickly with React Chart Zooming and Panning, you can check on this video:

Enable zooming

Chart can be zoomed in three ways.

- Selection - By setting [enableSelectionZooming](#)

property to true in `zoomSettings`, you can zoom the chart by using the rubber band selection.

- Mousewheel - By setting [enableMouseWheelZooming](#) property to true in `zoomSettings`, you can zoomin and zoomout the chart by scrolling the mouse wheel.
- Pinch - By setting [enablePinchZooming](#) property to true in `zoomSettings`, you can zoom the chart through pinch gesture in touch enabled devices.

Note: Pinch zooming is supported only in browsers that support multi-touch gestures. Currently IE11, Chrome and Opera browsers support multi-touch in desktop devices.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, AreaSeries, Zoom } from
"@syncfusion/ej2-react-charts";
import { zoomData } from "datasource.ts";
function App() {
  const primaryxAxis = { valueType: 'DateTime' };
  const legendSettings = { visible: false };
  const zoomsettings = {
    enableMouseWheelZooming: true, enablePinchZooming: true,
    enableSelectionZooming: true
  };
  const border = { width: 0.5, color: '#00bdae' };
  const animation = { enable: false };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
legendSettings={legendSettings} zoomSettings={zoomsettings} title='Sales
History of Product X'>
    <Inject services={[AreaSeries, Legend, Tooltip, DataLabel, Zoom,
DateTime]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={zoomData} xName='x' yName='y'
opacity={0.3} name='Product X' type='Area' border={border}
animation={animation}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel, ZoomSettingsModel,
Legend, DateTime, Tooltip, DataLabel, AreaSeries, Zoom}
from '@syncfusion/ej2-react-charts';
import { zoomData } from 'datasource.ts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime' };
  const legendSettings: LegendSettingsModel = { visible: false };
  const zoomsettings: ZoomSettingsModel = {
    enableMouseWheelZooming: true, enablePinchZooming: true,
    enableSelectionZooming: true
  };
  const border = { width: 0.5, color: '#00bdae' };
  const animation = { enable: false };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    legendSettings={legendSettings}
    zoomSettings={zoomsettings}
    title='Sales History of Product X'>
    <Inject services={[AreaSeries, Legend, Tooltip, DataLabel, Zoom,
DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={zoomData} xName='x' yName='y'
opacity={0.3} name='Product X' type='Area'
border={border} animation={animation}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
</App>
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

After zooming the chart, a zooming toolbar will appear with **zoom**, **zoomin**, **zoomout**, **pan** and **reset** buttons. Selecting the Pan option will allow to pan the chart and selecting the Reset option will reset the zoomed chart.

Modes

The [mode](#) property in zoomSettings specifies whether the chart is allowed to scale along the horizontal axis or vertical axis. The default value of the mode is XY (both axis).

There are three types of mode.

- X - Allows us to zoom the chart horizontally.
- Y - Allows us to zoom the chart vertically.
- XY - Allows us to zoom the chart both vertically and horizontally.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";

```

```

import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, AreaSeries, Zoom } from
'@syncfusion/ej2-react-charts';
import { zoomData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'DateTime' };
    const legendSettings = { visible: false };
    const zoomsettings = { mode: 'X', enableSelectionZooming: true };
    const border = { width: 0.5, color: '#00bdae' };
    const animation = { enable: false };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
legendSettings={legendSettings} zoomSettings={zoomsettings} title='Sales
History of Product X'>
        <Inject services={[AreaSeries, Legend, Tooltip, DataLabel, Zoom,
DateTime]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={zoomData} xName='x' yName='y'
opacity={0.3} name='Product X' type='Area' border={border}
animation={animation}>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
    }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel, ZoomSettingsModel,
Legend, DateTime, Tooltip, DataLabel, AreaSeries, Zoom }
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime' };
    const legendSettings: LegendSettingsModel = { visible: false };
    const zoomsettings: ZoomSettingsModel = { mode: 'X',
enableSelectionZooming: true };
    const border = { width: 0.5, color: '#00bdae' };
    const animation = { enable: false };
    const zoomData: Object[] = [
        { x: new Date(2016, 0, 1), y: 7.1 }, { x: new Date(2016, 1, 1), y: 3.7
    },
        { x: new Date(2016, 2, 1), y: 0.8 }, { x: new Date(2016, 3, 1), y: 6.3
    },
        { x: new Date(2016, 4, 1), y: 13.3 }, { x: new Date(2016, 5, 1), y: 18.0
    },
        { x: new Date(2016, 6, 1), y: 19.8 }, { x: new Date(2016, 7, 1), y: 18.1
    },
        { x: new Date(2016, 8, 1), y: 13.1 }, { x: new Date(2016, 9, 1), y: 4.1
    },
        { x: new Date(2016, 10, 1), y: -3.8 }, { x: new Date(2016, 11, 1), y: -
6.8 }
    ];

```

```

return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  legendSettings={legendSettings}
  zoomSettings={zoomsettings}
  title='Sales History of Product X'>
  <Inject services={[AreaSeries, Legend, Tooltip, DataLabel, Zoom,
DateTime]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={zoomData} xName='x' yName='y'
opacity={0.3} name='Product X' type='Area'
    border={border} animation={animation}>>
  </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Toolbar

By default, zoomin, zoomout, pan and reset buttons will be displayed for zoomed chart. You can customize to show the desired options in the toolbar using the [toolbarItems](#) property. Also using the [showToolbar](#) property, you can show toolkit for zooming and panning the chart during initial rendering itself.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, AreaSeries, Zoom } from
'@syncfusion/ej2-react-charts';
import { zoomData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'DateTime' };
  const legendSettings = { visible: false };
  const zoomsettings = { enableSelectionZooming: true, toolbarItems:
['Zoom', 'Pan', 'Reset'], showToolbar: true };
  const border = { width: 0.5, color: '#00bdae' };
  const animation = { enable: false };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
legendSettings={legendSettings} zoomSettings={zoomsettings} title='Sales
History of Product X'>
    <Inject services={[AreaSeries, Legend, Tooltip, DataLabel, Zoom,
DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={zoomData} xName='x' yName='y'
name='Product X' opacity={0.3} type='Area' border={border}
animation={animation}>>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel, ZoomSettingsModel,
Legend, DateTime, Tooltip, DataLabel, AreaSeries, Zoom}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime' };
    const legendSettings: LegendSettingsModel = { visible: false };
    const zoomsettings: ZoomSettingsModel = { enableSelectionZooming: true,
toolbarItems: ['Zoom', 'Pan', 'Reset'], showToolbar: true };
    const border = { width: 0.5, color: '#00bdae' };
    const animation = { enable: false };
    const zoomData: Object[] = [
        { x: new Date(2016, 0, 1), y: 7.1 }, { x: new Date(2016, 1, 1), y: 3.7
    },
        { x: new Date(2016, 2, 1), y: 0.8 }, { x: new Date(2016, 3, 1), y: 6.3
    },
        { x: new Date(2016, 4, 1), y: 13.3 }, { x: new Date(2016, 5, 1), y: 18.0
    },
        { x: new Date(2016, 6, 1), y: 19.8 }, { x: new Date(2016, 7, 1), y: 18.1
    },
        { x: new Date(2016, 8, 1), y: 13.1 }, { x: new Date(2016, 9, 1), y: 4.1
    },
        { x: new Date(2016, 10, 1), y: -3.8 }, { x: new Date(2016, 11, 1), y: -
6.8 }
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        legendSettings={legendSettings}
        zoomSettings={zoomsettings}
        title='Sales History of Product X'>
        <Inject services={[AreaSeries, Legend, Tooltip, DataLabel, Zoom,
DateTime]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={zoomData} xName='x' yName='y'
name='Product X' opacity={0.3} type='Area'
                border={border} animation={animation}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Enable scrollbar

Using the [enableScrollbar](#) property, you can add a scrollbar to a zoomed chart. This scrollbar allows you to zoom or pan the chart. The appearance of the scrollbar can be customized using properties in [scrollbarSettings](#). For example, you can use [trackColor](#) and [trackRadius](#) properties to customize the track of the scrollbar, and [scrollbarRadius](#) and [scrollbarColor](#) properties to customize the scroller. The ability to zoom through the scrollbar can be enabled or disabled using the [enableZoom](#) property in

[scrollbarSettings](#). Additionally, you can change the color of the grip and height of the scrollbar using the [gripColor](#) and [height](#) properties.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, AreaSeries, Zoom, ScrollBar } from
'@syncfusion/ej2-react-charts';
import { zoomData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'DateTime', zoomFactor: 0.2,
zoomPosition: 0.6, scrollbarSettings: { enable: true, enableZoom: false,
height: 14, trackRadius: 8, scrollbarRadius: 8, gripColor: 'transparent',
trackColor: 'yellow', scrollbarColor: 'red' } };
    const legendSettings = { visible: false };
    const zoomsettings = { enableSelectionZooming: true, enableScrollbar: true
};
    const border = { width: 0.5, color: '#00bdae' };
    const animation = { enable: false };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
legendSettings={legendSettings} zoomSettings={zoomsettings} title='Sales
History of Product X'>
        <Inject services={[AreaSeries, Legend, Tooltip, DataLabel, Zoom,
DateTime, ScrollBar]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={zoomData} xName='x' yName='y'
name='Product X' opacity={0.3} type='Area' border={border}
animation={animation}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
    AxisModel, ChartComponent, SeriesCollectionDirective, SeriesDirective,
    Inject, LegendSettingsModel, ZoomSettingsModel,
    Legend, DateTime, Tooltip, DataLabel, AreaSeries, Zoom, ScrollBar
}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime', zoomFactor: 0.2,
zoomPosition: 0.6, scrollbarSettings: { enable: true, enableZoom: false,
height: 14, trackRadius: 8, scrollbarRadius: 8, gripColor: 'transparent',
trackColor: 'yellow', scrollbarColor: 'red' } };
    const legendSettings = { visible: false };
    const zoomsettings = { enableSelectionZooming: true, enableScrollbar: true
};
    const border = { width: 0.5, color: '#00bdae' };
    const animation = { enable: false };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
legendSettings={legendSettings} zoomSettings={zoomsettings} title='Sales
History of Product X'>
        <Inject services={[AreaSeries, Legend, Tooltip, DataLabel, Zoom,
DateTime, ScrollBar]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={zoomData} xName='x' yName='y'
name='Product X' opacity={0.3} type='Area' border={border}
animation={animation}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

```

const legendSettings: LegendSettingsModel = { visible: false };
const zoomSettings: ZoomSettingsModel = { enableSelectionZooming: true,
enableScrollbar: true };
const border = { width: 0.5, color: '#00bdae' };
const animation = { enable: false };
const zoomData: Object[] = [
  { x: new Date(2016, 0, 1), y: 7.1 }, { x: new Date(2016, 1, 1), y: 3.7
},
  { x: new Date(2016, 2, 1), y: 0.8 }, { x: new Date(2016, 3, 1), y: 6.3
},
  { x: new Date(2016, 4, 1), y: 13.3 }, { x: new Date(2016, 5, 1), y: 18.0
},
  { x: new Date(2016, 6, 1), y: 19.8 }, { x: new Date(2016, 7, 1), y: 18.1
},
  { x: new Date(2016, 8, 1), y: 13.1 }, { x: new Date(2016, 9, 1), y: 4.1
},
  { x: new Date(2016, 10, 1), y: -3.8 }, { x: new Date(2016, 11, 1), y: -
6.8 }
];
return <ChartComponent id='charts'
  primaryXAxis={primaryXAxis}
  legendSettings={legendSettings}
  zoomSettings={zoomSettings}
  title='Sales History of Product X'>
  <Inject services={[AreaSeries, Legend, Tooltip, DataLabel, Zoom,
DateTime, ScrollBar]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={zoomData} xName='x' yName='y'
name='Product X' opacity={0.3} type='Area'
    border={border} animation={animation}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
</>
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Enable pan

Using [enablePan](#) property you can able to pan the zoomed chart without help of toolbar items.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, AreaSeries, Zoom } from
"@syncfusion/ej2-react-charts";
import { zoomData } from "datasource.ts";
function App() {
  const primaryXAxis = { valueType: 'DateTime', zoomFactor: 0.2,
zoomPosition: 0.6 };
  const legendSettings = { visible: false };
  const zoomSettings = { enableSelectionZooming: true, enablePan: true };
  const border = { width: 0.5, color: '#00bdae' };
  const animation = { enable: false };

```



```

    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
    legendSettings={legendSettings} zoomSettings={zoomSettings} title='Sales
    History of Product X'>
        <Inject services={[AreaSeries, Legend, Tooltip, DataLabel, Zoom,
    DateTime]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={zoomData} xName='x' yName='y'
    name='Product X' opacity={0.3} type='Area' border={border}
    animation={animation}>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
    }
    ;
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
    SeriesDirective, Inject, LegendSettingsModel, ZoomSettingsModel,
    Legend, DateTime, Tooltip, DataLabel, AreaSeries, Zoom }
    from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime', zoomFactor: 0.2,
    zoomPosition: 0.6 };
    const legendSettings: LegendSettingsModel = { visible: false };
    const zoomSettings: ZoomSettingsModel = { enableSelectionZooming: true,
    enablePan: true };
    const border = { width: 0.5, color: '#00bdae' };
    const animation = { enable: false };
    const zoomData: Object[] = [
        { x: new Date(2016, 0, 1), y: 7.1 }, { x: new Date(2016, 1, 1), y: 3.7
    },
        { x: new Date(2016, 2, 1), y: 0.8 }, { x: new Date(2016, 3, 1), y: 6.3
    },
        { x: new Date(2016, 4, 1), y: 13.3 }, { x: new Date(2016, 5, 1), y: 18.0
    },
        { x: new Date(2016, 6, 1), y: 19.8 }, { x: new Date(2016, 7, 1), y: 18.1
    },
        { x: new Date(2016, 8, 1), y: 13.1 }, { x: new Date(2016, 9, 1), y: 4.1
    },
        { x: new Date(2016, 10, 1), y: -3.8 }, { x: new Date(2016, 11, 1), y: -
    6.8 }
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        legendSettings={legendSettings}
        zoomSettings={zoomSettings}
        title='Sales History of Product X'>
        <Inject services={[AreaSeries, Legend, Tooltip, DataLabel, Zoom,
    DateTime]}/>
        <SeriesCollectionDirective>

```

```

        <SeriesDirective dataSource={zoomData} xName='x' yName='y'
name='Product X' opacity={0.3} type='Area'
        border={border} animation={animation}>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Auto interval on zooming

By using [enableAutoIntervalOnZooming](#) property, the axis interval will get calculated automatically with respect to the zoomed range.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, AreaSeries, Zoom } from
'@syncfusion/ej2-react-charts';
import { zoomData } from 'datasource.ts';
function App() {
    const primaryXAxis = { valueType: 'DateTime',
enableAutoIntervalOnZooming: true };
    const legendSettings = { visible: false };
    const zoomSettings = { enableSelectionZooming: true };
    const border = { width: 0.5, color: '#00bdae' };
    const animation = { enable: false };
    return <ChartComponent id='charts' primaryXAxis={primaryXAxis}
legendSettings={legendSettings} zoomSettings={zoomSettings} title='Sales
History of Product X'>
        <Inject services={[AreaSeries, Legend, Tooltip, DataLabel, Zoom,
DateTime]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={zoomData} xName='x' yName='y'
name='Product X' opacity={0.3} type='Area' border={border}
animation={animation}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, LegendSettingsModel, ZoomSettingsModel,
Legend, DateTime, Tooltip, DataLabel, AreaSeries, Zoom}
from '@syncfusion/ej2-react-charts';
function App() {

```

```

const primaryxAxis: AxisModel = { valueType: 'DateTime',
enableAutoIntervalOnZooming: true };
const legendSettings: LegendSettingsModel = { visible: false };
const zoomSettings: ZoomSettingsModel = { enableSelectionZooming: true };
const border = { width: 0.5, color: '#00bdae' };
const animation = { enable: false };
const zoomData: Object[] = [
  { x: new Date(2016, 0, 1), y: 7.1 }, { x: new Date(2016, 1, 1), y: 3.7
},
  { x: new Date(2016, 2, 1), y: 0.8 }, { x: new Date(2016, 3, 1), y: 6.3
},
  { x: new Date(2016, 4, 1), y: 13.3 }, { x: new Date(2016, 5, 1), y: 18.0
},
  { x: new Date(2016, 6, 1), y: 19.8 }, { x: new Date(2016, 7, 1), y: 18.1
},
  { x: new Date(2016, 8, 1), y: 13.1 }, { x: new Date(2016, 9, 1), y: 4.1
},
  { x: new Date(2016, 10, 1), y: -3.8 }, { x: new Date(2016, 11, 1), y: -
6.8 }
];
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  legendSettings={legendSettings}
  zoomSettings={zoomSettings}
  title='Sales History of Product X'>
  <Inject services={[AreaSeries, Legend, Tooltip, DataLabel, Zoom,
DateTime]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={zoomData} xName='x' yName='y'
name='Product X' opacity={0.3} type='Area'
    border={border} animation={animation}>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Note: To use zooming feature, we need to inject **Zoom** module into the **services**.

<!-- markdownlint-disable MD036 -->

Data editing in React Chart component

Enable Data Editing

We can use the data editing through inject the **DataEditing** module in the chart. It provides drag and drop support to the rendered points. Now, we can change the location or value of the point based on its **y** value. To enable the data editing, set the **enable** property to true in the drag settings of the series. Also, we can set color using **fill** property and set the data editing minimum and maximum range using **minY** and **maxY** properties.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";

```

```

import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries, LineSeries, DataEditing
} from '@syncfusion/ej2-react-charts';
function App() {
    const columnData = [
        { x: '2005', y: 21 }, { x: '2006', y: 60 },
        { x: '2007', y: 45 }, { x: '2008', y: 50 },
        { x: '2009', y: 74 }, { x: '2010', y: 65 },
        { x: '2011', y: 85 }
    ];
    const lineData = [
        { x: '2005', y: 21 }, { x: '2006', y: 22 },
        { x: '2007', y: 36 }, { x: '2008', y: 34 },
        { x: '2009', y: 54 }, { x: '2010', y: 55 },
        { x: '2011', y: 60 }
    ];
    const primaryxAxis = { valueType: 'Category', minimum: -0.5, maximum:
6.5, labelPlacement: 'OnTicks', majorGridLines: { width: 0 } };
    const primaryyAxis = { rangePadding: 'None', minimum: 0, title: 'Sales',
labelFormat: '{value}%', maximum: 100, interval: 20, lineStyle: { width: 0
}, majorTickLines: { width: 0 }, minorTickLines: { width: 0 } };
    const border = { border: { width: 0 } };
    const tooltip = { enable: true };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} chartArea={border} title='Sales Prediction of
Products' tooltip={tooltip}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
LineSeries, DataEditing]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={columnData} xName='x' yName='y'
name='Product A' type='Column' dragSettings={{ enable: true }} marker={{
visible: true, width: 10, height: 10 }}>
            </SeriesDirective>
            <SeriesDirective dataSource={lineData} xName='x' yName='y'
name='Product B' type='Line' marker={{ visible: true, width: 10, height: 10
}} dragSettings={{ enable: true }}></SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    Legend, Category, Tooltip, DataLabel, ColumnSeries, LineSeries,
DataEditing, TooltipSettingsModel}
from '@syncfusion/ej2-react-charts';
function App() {
    const columnData: any[] = [

```

```

        { x: '2005', y: 21 }, { x: '2006', y: 60 },
        { x: '2007', y: 45 }, { x: '2008', y: 50 },
        { x: '2009', y: 74 }, { x: '2010', y: 65 },
        { x: '2011', y: 85 }
    ];
    const lineData: any[] = [
        { x: '2005', y: 21 }, { x: '2006', y: 22 },
        { x: '2007', y: 36 }, { x: '2008', y: 34 },
        { x: '2009', y: 54 }, { x: '2010', y: 55 },
        { x: '2011', y: 60 }
    ];
    const primaryxAxis: AxisModel = { valueType: 'Category', minimum: -0.5,
    maximum: 6.5, labelPlacement: 'OnTicks', majorGridLines: { width: 0 } };
    const primaryyAxis: AxisModel = { rangePadding: 'None', minimum: 0,
    title: 'Sales', labelFormat: '{value}%', maximum: 100, interval: 20,
    lineStyle: { width: 0 }, majorTickLines: { width: 0 }, minorTickLines: {
    width: 0 } };
    const border = { border: { width: 0 } };
    const tooltip: TooltipSettingsModel = { enable: true };
    return <ChartComponent id='charts' primaryXAxis={ primaryxAxis }
    primaryYAxis={ primaryyAxis } chartArea={border} title='Sales Prediction of
    Products' tooltip={tooltip}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Category,
    LineSeries, DataEditing ]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={columnData} xName='x' yName='y'
            name='Product A' type='Column' dragSettings={{enable: true}}
            marker={{visible: true, width: 10, height: 10}}>
            </SeriesDirective>
            <SeriesDirective dataSource={lineData} xName='x' yName='y'
            name='Product B' type='Line' marker={{visible: true, width: 10, height: 10}}
            dragSettings={{enable: true}}></SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
    };
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));
    {% enddraw %}

```

Cross hair and track ball in React Chart component

Crosshair has a vertical and horizontal line to view the value of the axis at mouse or touch position.

Crosshair lines can be enabled by using [enable](#) property in the `crosshair`. Likewise tooltip label for an axis can be enabled by using [enable](#) property of `crosshairTooltip` in the corresponding axis.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
    Legend, DateTime, Tooltip, DataLabel, LineSeries, Crosshair } from
    '@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'DateTime' };
    const crosshair = { enable: true };

```

```

    const marker = { visible: true };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
crosshair={crosshair} title='Sales History of Product X'>
      <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Crosshair,
DateTime]}/>
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y' name='Product
X' type='Line' marker={marker}>
          </SeriesDirective>
        </SeriesCollectionDirective>
      </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, CrosshairSettingsModel,
Legend, DateTime, Tooltip, DataLabel, LineSeries, Crosshair}
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime' };
  const crosshair: CrosshairSettingsModel = { enable: true };
  const marker = { visible: true };
  const data: any[] = [
    { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
    { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
    { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
    { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
    { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
    { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
    { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
    { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    crosshair={crosshair}
    title='Sales History of Product X'>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Crosshair,
DateTime]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' name='Product
X' type='Line'
        marker={marker} >
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Tooltip for axis

Tooltip label for an axis can be enabled by using [enable](#) property of `crosshairTooltip` in the corresponding axis.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, LineSeries, Crosshair } from
'@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'DateTime', crosshairTooltip: {
enable: true } };
    const primaryyAxis = { crosshairTooltip: { enable: true } };
    const crosshair = { enable: true };
    const marker = { visible: true };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} crosshair={crosshair} title='Sales History of
Product X'>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Crosshair,
DateTime]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' name='Product
X' type='Line' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, CrosshairSettingsModel,
Legend, DateTime, Tooltip, DataLabel, LineSeries, Crosshair}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'DateTime', crosshairTooltip:
{ enable: true } };
    const primaryyAxis: AxisModel = { crosshairTooltip: { enable: true } };
    const crosshair: CrosshairSettingsModel = { enable: true };
    const marker = { visible: true };
    const data: any[] = [
        { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
        { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
        { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
        { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
        { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
        { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
        { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
    ];
```

```

    { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    crosshair={crosshair}
    title='Sales History of Product X'>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Crosshair,
    DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' name='Product
    X' type='Line'
        marker={marker} >
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Customization

The [fill](#) and [textStyle](#) property of the [crosshairTooltip](#) is used to customize the background color and font style of the crosshair label respectively. Color and width of the crosshair line can be customized by using the [line](#) property in the crosshair.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, LineSeries, Crosshair } from
'@syncfusion/ej2-react-charts';
import { data } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'DateTime', crosshairTooltip: {
enable: true, fill: 'green' } };
  const primaryyAxis = { crosshairTooltip: { enable: true, fill: 'green' }
};
  const crosshair = { enable: true, line: { width: 2, color: 'green' } };
  const marker = { visible: true };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} crosshair={crosshair} title='Sales History of
Product X'>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Crosshair,
    DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' name='Product
    X' type='Line' marker={marker}>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y1'
name='Product X' type='Line' marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;

```



```
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, CrosshairSettingsModel,
Legend, DateTime, Tooltip, DataLabel, LineSeries, Crosshair }
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'DateTime', crosshairTooltip:
{ enable: true, fill: 'green' } };
  const primaryyAxis: AxisModel = { crosshairTooltip: { enable: true, fill:
'green' } };
  const crosshair: CrosshairSettingsModel = { enable: true, line: { width:
2, color: 'green' } };
  const marker = { visible: true };
  const data: any[] = [
    { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
    { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
    { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
    { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
    { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
    { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
    { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
    { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    crosshair={crosshair}
    title='Sales History of Product X'>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Crosshair,
DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' name='Product
X' type='Line'
        marker={marker} >
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y1'
name='Product X' type='Line'
        marker={marker} >
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Note: To use crosshair feature, we need to inject **Crosshair** module into the **services**.

Trackball

Trackball is used to track a data point closest to the mouse or touch position. Trackball marker indicates the closest point and trackball tooltip displays the information about the point. To use trackball feature, we need to inject **Crosshair** and **Tooltip** module into the **services**.

Trackball can be enabled by setting the [enable](#) property of the crosshair to true and [shared](#) property in **tooltip** to true in chart.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, LineSeries, Crosshair } from
'@syncfusion/ej2-react-charts';
import { trackData } from 'datasource.ts';
function App() {
    const primaryxAxis = {
        title: 'Years', minimum: new Date(2000, 1, 1),
        maximum: new Date(2006, 2, 11), intervalType: 'Years', valueType:
'DateTime'
    };
    const primaryyAxis = { crosshairTooltip: { enable: true, fill: 'green' }
};
    const crosshair = { enable: true, lineType: 'Vertical' };
    const marker = { visible: true };
    const tooltip = { enable: true, shared: true, format: '${series.name} :
${point.x} : ${point.y}' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
crosshair={crosshair} tooltip={tooltip} title='Average Sales per Person'>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Crosshair,
DateTime]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={trackData} xName='x' yName='y'
name='John' type='Line' width={2} marker={marker}>
            </SeriesDirective>
            <SeriesDirective dataSource={trackData} xName='x' yName='y1'
name='Andrew' type='Line' width={2} marker={marker}></SeriesDirective>
            <SeriesDirective dataSource={trackData} xName='x' yName='y2'
name='Thomas' type='Line' width={2} marker={marker}></SeriesDirective>
            <SeriesDirective dataSource={trackData} xName='x' yName='y3'
name='Mark' type='Line' width={2} marker={marker}></SeriesDirective>
            <SeriesDirective dataSource={trackData} xName='x' yName='y4'
name='William' type='Line' width={2} marker={marker}></SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
```

```

import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, CrosshairSettingsModel, TooltipSettingsModel,
Legend, DateTime, Tooltip, DataLabel, LineSeries, Crosshair}
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = {
    title: 'Years', minimum: new Date(2000, 1, 1),
    maximum: new Date(2006, 2, 11), intervalType: 'Years', valueType:
'DateTime'
  };
  const primaryyAxis: AxisModel = { crosshairTooltip: { enable: true, fill:
'green' } };
  const crosshair: CrosshairSettingsModel = { enable: true, lineType:
'Vertical' };
  const marker = { visible: true };
  const tooltip: TooltipSettingsModel = { enable: true, shared: true,
format: '${series.name} : ${point.x} : ${point.y}' };
  const trackData: any[] = [
    { x: new Date(2000, 2, 11), y: 15, y1: 39, y2: 60, y3: 75, y4: 85 },
    { x: new Date(2000, 9, 14), y: 20, y1: 30, y2: 55, y3: 75, y4: 83 },
    { x: new Date(2001, 2, 11), y: 25, y1: 28, y2: 48, y3: 68, y4: 85 },
    { x: new Date(2001, 9, 16), y: 21, y1: 35, y2: 57, y3: 75, y4: 87 },
    { x: new Date(2002, 2, 7), y: 13, y1: 39, y2: 62, y3: 71, y4: 82 },
    { x: new Date(2002, 9, 7), y: 18, y1: 41, y2: 64, y3: 69, y4: 74 },
    { x: new Date(2003, 2, 11), y: 24, y1: 45, y2: 57, y3: 81, y4: 73 },
    { x: new Date(2003, 9, 14), y: 23, y1: 48, y2: 53, y3: 84, y4: 75 },
    { x: new Date(2004, 2, 6), y: 19, y1: 54, y2: 63, y3: 85, y4: 73 },
    { x: new Date(2004, 9, 6), y: 31, y1: 55, y2: 50, y3: 87, y4: 60 },
    { x: new Date(2005, 2, 11), y: 39, y1: 57, y2: 66, y3: 75, y4: 48 },
    { x: new Date(2005, 9, 11), y: 50, y1: 60, y2: 65, y3: 70, y4: 55 },
    { x: new Date(2006, 2, 11), y: 24, y1: 60, y2: 79, y3: 85, y4: 40 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    crosshair={crosshair}
    tooltip={tooltip}
    title='Average Sales per Person'>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Crosshair,
DateTime]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={trackData} xName='x' yName='y'
name='John' type='Line' width={2}
        marker={marker} >
      </SeriesDirective>
      <SeriesDirective dataSource={trackData} xName='x' yName='y1'
name='Andrew' type='Line' width={2}
        marker={marker} ></SeriesDirective>
      <SeriesDirective dataSource={trackData} xName='x' yName='y2'
name='Thomas' type='Line' width={2}
        marker={marker} ></SeriesDirective>
      <SeriesDirective dataSource={trackData} xName='x' yName='y3'
name='Mark' type='Line' width={2}
        marker={marker} ></SeriesDirective>
      <SeriesDirective dataSource={trackData} xName='x' yName='y4'
name='William' type='Line' width={2}
        marker={marker} ></SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>

```

```

    </ChartComponent>
  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));

```

Synchronized Charts in React Chart component

Tooltip synchronization

The tooltip can be synchronized across multiple charts using the [showTooltip](#) and [hideTooltip](#) methods. When we hover over a data point in one chart, we call the [showTooltip](#) method for the other charts to display related information in other connected charts simultaneously.

In the [showTooltip](#) method, specify the following parameters programmatically to enable tooltip for a particular chart:

- **x** - Data point x-value or x-coordinate value.
- **y** - Data point y-value or y-coordinate value.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { Chart, AreaSeries, LineSeries, DateTime, Tooltip, IMouseEventArgs,
ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject } from
'@syncfusion/ej2-react-charts';
import { synchronizedData } from 'datasource.ts';
import { Browser } from '@syncfusion/ej2-base';
function App() {
  let chart1;
  let chart2;
  let chart1MouseLeave = (args) => {
    chart2.hideTooltip();
  };
  let chart1MouseMove = (args) => {
    if (!!Browser.isDevice && !chart1.isTouch && !chart1.isChartDrag) ||
chart1.startMove) {
      chart2.startMove = chart1.startMove;
      chart2.showTooltip(args.x, args.y);
    }
  };
  let chart1MouseUp = (args) => {
    if (Browser.isDevice && chart1.startMove) {
      chart2.hideTooltip();
    }
  };
  let chart2MouseLeave = (args) => {
    chart1.hideTooltip();
  };
  let chart2MouseMove = (args) => {
    if (!!Browser.isDevice && !chart2.isTouch && !chart2.isChartDrag) ||
chart2.startMove) {
      chart1.startMove = chart2.startMove;
      chart1.showTooltip(args.x, args.y);
    }
  };

```

```

    }
  };
  let chart2MouseUp = (args) => {
    if (Browser.isDevice && chart2.startMove) {
      chart1.hideTooltip();
    }
  };
  return <div className="control-section">
    <div className="row">
      <div className="col">
        <ChartComponent
          id="container1"
          ref={chart => chart1 = chart}
          primaryXAxis={{
            minimum: new Date(2023, 1, 18),
            maximum: new Date(2023, 7, 18),
            valueType: 'DateTime',
            labelFormat: 'MMM d',
            lineStyle: { width: 0 },
            majorGridLines: { width: 0 },
            edgeLabelPlacement: Browser.isDevice ? 'None' : 'Shift',
            labelRotation: Browser.isDevice ? -45 : 0,
            interval: Browser.isDevice ? 2 : 1
          }}
          primaryYAxis={{
            labelFormat: 'n2',
            majorTickLines: { width: 0 },
            lineStyle: { width: 0 },
            minimum: 0.86,
            maximum: 0.96,
            interval: 0.025
          }}
          chartArea={{ border: { width: 0 } }}
          chartMouseLeave={chart1MouseLeave.bind(this)}
          chartMouseMove={chart1MouseMove.bind(this)}
          chartMouseUp={chart1MouseUp.bind(this)}
          titleStyle={{ textAlign: 'Near' }}
          tooltip={{ enable: true, fadeOutDuration: Browser.isDevice ? 2500
: 1000, shared: true, header: '', format: '<b>€${point.y}</b><br>${point.x}
2023', enableMarker: false }}
          title="US to Euro">
            <Inject services={[LineSeries, DateTime, Tooltip]} />
            <SeriesCollectionDirective>
              <SeriesDirective type="Line" dataSource={synchronizedData}
xName="USD" yName="EUR" width={2} emptyPointSettings={{ mode: 'Drop'
}}></SeriesDirective>
            </SeriesCollectionDirective>
          </ChartComponent>
        </div>
        <div className="col">
          <ChartComponent
            id="container2"
            ref={chart => chart2 = chart}
            primaryXAxis={{
              minimum: new Date(2023, 1, 18),
              maximum: new Date(2023, 7, 18),
              valueType: 'DateTime',

```

```

        labelFormat: 'MMM d',
        lineStyle: { width: 0 },
        majorGridLines: { width: 0 },
        edgeLabelPlacement: Browser.isDevice ? 'None' : 'Shift',
        labelRotation: Browser.isDevice ? -45 : 0,
        interval: Browser.isDevice ? 2 : 1
    })
    primaryYAxis={({
        labelFormat: 'n1',
        majorTickLines: { width: 0 },
        lineStyle: { width: 0 },
        minimum: 79,
        maximum: 85,
        interval: 1.5
    })
    chartArea={{ border: { width: 0 } }}
    chartMouseLeave={chart2MouseLeave.bind(this)}
    chartMouseMove={chart2MouseMove.bind(this)}
    chartMouseUp={chart2MouseUp.bind(this)}
    titleStyle={{ textAlign: 'Near' }}
    tooltip={{ enable: true, fadeOutDuration: Browser.isDevice ? 2500
: 1000, shared: true, header: '', format: '<b>₹${point.y}</b><br>${point.x}
2023', enableMarker: false }}
    title="US to INR">
    <Inject services={[AreaSeries, DateTime, Tooltip]} />
    <SeriesCollectionDirective>
        <SeriesDirective type="Area" dataSource={synchronizedData}
xName="USD" yName="INR" opacity={0.6} width={2} border={{ width: 2
}}></SeriesDirective>
    </SeriesCollectionDirective>
    </ChartComponent>
</div>
</div>
</div>
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { Chart, AreaSeries, LineSeries, DateTime, Tooltip, IMouseEventArgs,
ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject } from
'@syncfusion/ej2-react-charts';
import { synchronizedData } from 'datasource.ts';
import { Browser } from '@syncfusion/ej2-base';
function App() {
    let chart1: ChartComponent;
    let chart2: ChartComponent;
    let chart1MouseLeave = (args: IMouseEventArgs): void => {
        chart2.hideTooltip();
    };

```

```

let chart1MouseMove = (args: IMouseEventArgs): void => {
    if ((!Browser.isDevice && !chart1.isTouch && !chart1.isChartDrag) ||
chart1.startMove) {
        chart2.startMove = chart1.startMove;
        chart2.showTooltip(args.x, args.y);
    }
};
let chart1MouseUp = (args: IMouseEventArgs): void => {
    if (Browser.isDevice && chart1.startMove) {
        chart2.hideTooltip();
    }
};
let chart2MouseLeave = (args: IMouseEventArgs): void => {
    chart1.hideTooltip();
};
let chart2MouseMove = (args: IMouseEventArgs): void => {
    if ((!Browser.isDevice && !chart2.isTouch && !chart2.isChartDrag) ||
chart2.startMove) {
        chart1.startMove = chart2.startMove;
        chart1.showTooltip(args.x, args.y);
    }
};
let chart2MouseUp = (args: IMouseEventArgs): void => {
    if (Browser.isDevice && chart2.startMove) {
        chart1.hideTooltip();
    }
}
return <div className="control-section">
    <div className="row">
        <div className="col">
            <ChartComponent
                id="container1"
                ref={chart => chart1 = chart}
                primaryXAxis={{
                    minimum: new Date(2023, 1, 18),
                    maximum: new Date(2023, 7, 18),
                    valueType: 'DateTime',
                    labelFormat: 'MMM d',
                    lineStyle: { width: 0 },
                    majorGridLines: { width: 0 },
                    edgeLabelPlacement: Browser.isDevice ? 'None' : 'Shift',
                    labelRotation: Browser.isDevice ? -45 : 0,
                    interval: Browser.isDevice ? 2 : 1
                }}
                primaryYAxis={{
                    labelFormat: 'n2',
                    majorTickLines: { width: 0 },
                    lineStyle: { width: 0 },
                    minimum: 0.86,
                    maximum: 0.96,
                    interval: 0.025
                }}
                chartArea={{ border: { width: 0 } }}
                chartMouseLeave={chart1MouseLeave.bind(this)}
                chartMouseMove={chart1MouseMove.bind(this)}
                chartMouseUp={chart1MouseUp.bind(this)}
                titleStyle={{ textAlign: 'Near' }}

```

```

        tooltip={{ enable: true, fadeOutDuration: Browser.isDevice ? 2500
: 1000, shared: true, header: '', format: '<b>€${point.y}</b><br>${point.x}
2023', enableMarker: false }}
        title="US to Euro">
        <Inject services={[LineSeries, DateTime, Tooltip]} />
        <SeriesCollectionDirective>
            <SeriesDirective type="Line" dataSource={synchronizedData}
xName="USD" yName="EUR" width={2} emptyPointSettings={{ mode: 'Drop'
}}></SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
</div>
<div className="col">
    <ChartComponent
        id="container2"
        ref={chart => chart2 = chart}
        primaryXAxis={{
            minimum: new Date(2023, 1, 18),
            maximum: new Date(2023, 7, 18),
            valueType: 'DateTime',
            labelFormat: 'MMM d',
            lineStyle: { width: 0 },
            majorGridLines: { width: 0 },
            edgeLabelPlacement: Browser.isDevice ? 'None' : 'Shift',
            labelRotation: Browser.isDevice ? -45 : 0,
            interval: Browser.isDevice ? 2 : 1
        }}
        primaryYAxis={{
            labelFormat: 'n1',
            majorTickLines: { width: 0 },
            lineStyle: { width: 0 },
            minimum: 79,
            maximum: 85,
            interval: 1.5
        }}
        chartArea={{ border: { width: 0 } }}
        chartMouseLeave={chart2MouseLeave.bind(this)}
        chartMouseMove={chart2MouseMove.bind(this)}
        chartMouseUp={chart2MouseUp.bind(this)}
        titleStyle={{ textAlign: 'Near' }}
        tooltip={{ enable: true, fadeOutDuration: Browser.isDevice ? 2500
: 1000, shared: true, header: '', format: '<b>₹${point.y}</b><br>${point.x}
2023', enableMarker: false }}
        title="US to INR">
        <Inject services={[AreaSeries, DateTime, Tooltip]} />
        <SeriesCollectionDirective>
            <SeriesDirective type="Area" dataSource={synchronizedData}
xName="USD" yName="INR" opacity={0.6} width={2} border={{ width: 2
}}></SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
</div>
</div>
</div>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```



```
{% enddraw %}
```

Crosshair synchronization

The crosshair can be synchronized across multiple charts using the [showCrosshair](#) and [hideCrosshair](#) methods. When we hover over one chart, we call the [showCrosshair](#) method for the other charts to align with data points in other connected charts, simplifying data comparison and analysis.

In the [showCrosshair](#) method, specify the following parameters programmatically to enable crosshair for a particular chart:

- **x** - Specifies the x-value of the point or x-coordinate.
- **y** - Specifies the y-value of the point or y-coordinate.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { Chart, AreaSeries, SplineSeries, DateTime, Crosshair,
IMouseEventArgs, ChartComponent, SeriesCollectionDirective, SeriesDirective,
Inject } from '@syncfusion/ej2-react-charts';
import { synchronizedData } from 'datasource.ts';
import { Browser } from '@syncfusion/ej2-base';
function App() {
  let chart1;
  let chart2;
  let chart1MouseLeave = (args) => {
    chart2.hideCrosshair();
  };
  let chart1MouseMove = (args) => {
    if (!!Browser.isDevice && !chart1.isTouch && !chart1.isChartDrag) ||
chart1.startMove) {
      chart2.startMove = chart1.startMove;
      chart2.showCrosshair(args.x, args.y);
    }
  };
  let chart1MouseUp = (args) => {
    if (Browser.isDevice && chart1.startMove) {
      chart2.hideCrosshair();
    }
  };
  let chart2MouseLeave = (args) => {
    chart1.hideCrosshair();
  };
  let chart2MouseMove = (args) => {
    if (!!Browser.isDevice && !chart2.isTouch && !chart2.isChartDrag) ||
chart2.startMove) {
      chart1.startMove = chart2.startMove;
      chart1.showCrosshair(args.x, args.y);
    }
  };
  let chart2MouseUp = (args) => {
    if (Browser.isDevice && chart2.startMove) {
      chart1.hideCrosshair();
    }
  };
}
```

```

    }
    };
    return <div className="control-section">
      <div className="row">
        <div className="col">
          <ChartComponent
            id="container1"
            ref={chart => chart1 = chart}
            primaryXAxis={{
              minimum: new Date(2023, 1, 18),
              maximum: new Date(2023, 7, 18),
              valueType: 'DateTime',
              labelFormat: 'MMM d',
              lineStyle: { width: 0 },
              majorGridLines: { width: 0 },
              edgeLabelPlacement: Browser.isDevice ? 'None' :
'Shift',

              labelRotation: Browser.isDevice ? -45 : 0,
              interval: Browser.isDevice ? 2 : 1,
              crosshairTooltip: { enable: true }
            }}
            primaryYAxis={{
              labelFormat: 'n2',
              majorTickLines: { width: 0 },
              lineStyle: { width: 0 },
              minimum: 0.86,
              maximum: 0.96,
              interval: 0.025
            }}
            chartArea={{ border: { width: 0 } }}
            chartMouseLeave={chart1MouseLeave.bind(this)}
            chartMouseMove={chart1MouseMove.bind(this)}
            chartMouseUp={chart1MouseUp.bind(this)}
            titleStyle={{ textAlign: 'Near' }}
            crosshair={{ enable: true, lineType: 'Vertical',
dashArray: '2,2' }}
            title="US to Euro">
            <Inject services={[SplineSeries, DateTime, Crosshair]}
            />

            <SeriesCollectionDirective>
              <SeriesDirective type="Spline"
dataSource={synchronizedData} xName="USD" yName="EUR" width={2}
emptyPointSettings={{ mode: 'Drop' }}></SeriesDirective>
            </SeriesCollectionDirective>
          </ChartComponent>
        </div>
        <div className="col">
          <ChartComponent
            id="container2"
            ref={chart => chart2 = chart}
            primaryXAxis={{
              minimum: new Date(2023, 1, 18),
              maximum: new Date(2023, 7, 18),
              valueType: 'DateTime',
              labelFormat: 'MMM d',
              lineStyle: { width: 0 },
              majorGridLines: { width: 0 },

```

```

        edgeLabelPlacement: Browser.isDevice ? 'None' :
'Shift',
        labelRotation: Browser.isDevice ? -45 : 0,
        interval: Browser.isDevice ? 2 : 1,
        crosshairTooltip: { enable: true }
    }}
    primaryYAxis={{
        labelFormat: 'n1',
        majorTickLines: { width: 0 },
        lineStyle: { width: 0 },
        minimum: 79,
        maximum: 85,
        interval: 1.5
    }}
    chartArea={{ border: { width: 0 } }}
    chartMouseLeave={chart2MouseLeave.bind(this)}
    chartMouseMove={chart2MouseMove.bind(this)}
    chartMouseUp={chart2MouseUp.bind(this)}
    titleStyle={{ textAlign: 'Near' }}
    crosshair={{ enable: true, lineType: 'Vertical',
dashArray: '2,2' }}
    title="US to INR">
<Inject services={[AreaSeries, DateTime, Crosshair]} />
<SeriesCollectionDirective>
    <SeriesDirective type="Area"
dataSource={synchronizedData} xName="USD" yName="INR" opacity={0.6}
width={2} border={{ width: 2 }}></SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
</div>
</div>
</div>
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { Chart, AreaSeries, SplineSeries, DateTime, Crosshair,
IMouseEventArgs, ChartComponent, SeriesCollectionDirective, SeriesDirective,
Inject } from '@syncfusion/ej2-react-charts';
import { synchronizedData } from 'datasource.ts';
import { Browser } from '@syncfusion/ej2-base';
function App() {
    let chart1: ChartComponent;
    let chart2: ChartComponent;
    let chart1MouseLeave = (args: IMouseEventArgs): void => {
        chart2.hideCrosshair();
    };
    let chart1MouseMove = (args: IMouseEventArgs): void => {

```

```

        if ((!Browser.isDevice && !chart1.isTouch && !chart1.isChartDrag) ||
chart1.startMove) {
            chart2.startMove = chart1.startMove;
            chart2.showCrosshair(args.x, args.y);
        }
    };
    let chart1MouseUp = (args: IMouseEventArgs): void => {
        if (Browser.isDevice && chart1.startMove) {
            chart2.hideCrosshair();
        }
    };
    let chart2MouseLeave = (args: IMouseEventArgs): void => {
        chart1.hideCrosshair();
    };
    let chart2MouseMove = (args: IMouseEventArgs): void => {
        if ((!Browser.isDevice && !chart2.isTouch && !chart2.isChartDrag) ||
chart2.startMove) {
            chart1.startMove = chart2.startMove;
            chart1.showCrosshair(args.x, args.y);
        }
    };
    let chart2MouseUp = (args: IMouseEventArgs): void => {
        if (Browser.isDevice && chart2.startMove) {
            chart1.hideCrosshair();
        }
    }
}
return <div className="control-section">
    <div className="row">
        <div className="col">
            <ChartComponent
                id="container1"
                ref={chart => chart1 = chart}
                primaryXAxis={{
                    minimum: new Date(2023, 1, 18),
                    maximum: new Date(2023, 7, 18),
                    valueType: 'DateTime',
                    labelFormat: 'MMM d',
                    lineStyle: { width: 0 },
                    majorGridLines: { width: 0 },
                    edgeLabelPlacement: Browser.isDevice ? 'None' :
'Shift',

                    labelRotation: Browser.isDevice ? -45 : 0,
                    interval: Browser.isDevice ? 2 : 1,
                    crosshairTooltip: { enable: true }
                }}
                primaryYAxis={{
                    labelFormat: 'n2',
                    majorTickLines: { width: 0 },
                    lineStyle: { width: 0 },
                    minimum: 0.86,
                    maximum: 0.96,
                    interval: 0.025
                }}
                chartArea={{ border: { width: 0 } }}
                chartMouseLeave={chart1MouseLeave.bind(this)}
                chartMouseMove={chart1MouseMove.bind(this)}
                chartMouseUp={chart1MouseUp.bind(this)}

```

```

dashArray: '2,2' }}
crosshair={{ enable: true, lineType: 'Vertical',
titleStyle={{ textAlign: 'Near' }}
title="US to Euro">
<Inject services={[SplineSeries, DateTime, Crosshair]}
/>

<SeriesCollectionDirective>
  <SeriesDirective type="Spline"
dataSource={synchronizedData} xName="USD" yName="EUR" width={2}
emptyPointSettings={{ mode: 'Drop' }}></SeriesDirective>
</SeriesCollectionDirective>
</ChartComponent>
</div>
<div className="col">
  <ChartComponent
    id="container2"
    ref={chart => chart2 = chart}
    primaryXAxis={{
      minimum: new Date(2023, 1, 18),
      maximum: new Date(2023, 7, 18),
      valueType: 'DateTime',
      labelFormat: 'MMM d',
      lineStyle: { width: 0 },
      majorGridLines: { width: 0 },
      edgeLabelPlacement: Browser.isDevice ? 'None' :
'Shift',

      labelRotation: Browser.isDevice ? -45 : 0,
      interval: Browser.isDevice ? 2 : 1,
      crosshairTooltip: { enable: true }
    }}
    primaryYAxis={{
      labelFormat: 'n1',
      majorTickLines: { width: 0 },
      lineStyle: { width: 0 },
      minimum: 79,
      maximum: 85,
      interval: 1.5
    }}
    chartArea={{ border: { width: 0 } }}
    chartMouseLeave={chart2MouseLeave.bind(this)}
    chartMouseMove={chart2MouseMove.bind(this)}
    chartMouseUp={chart2MouseUp.bind(this)}
    crosshair={{ enable: true, lineType: 'Vertical',
dashArray: '2,2' }}
    titleStyle={{ textAlign: 'Near' }}
    title="US to INR">
<Inject services={[AreaSeries, DateTime, Crosshair]} />
<SeriesCollectionDirective>
  <SeriesDirective type="Area"
dataSource={synchronizedData} xName="USD" yName="INR" opacity={0.6}
width={2} border={{ width: 2 }}></SeriesDirective>
</SeriesCollectionDirective>
</ChartComponent>
</div>
</div>
</div>
};

```

```
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

Zooming synchronization

You can maintain constant zoom levels across multiple charts using the [zoomComplete](#) event. In the [zoomComplete](#) event, obtain the [zoomFactor](#) and [zoomPosition](#) values of the particular chart, and then apply those values to the other charts.

INDEX.JSX

```
{% raw %}
import * as ReactDOM from "react-dom";
import { Chart, SplineAreaSeries, LineSeries, DateTime, Zoom,
IZoomCompleteEventArgs, Selection, ChartComponent,
SeriesCollectionDirective, SeriesDirective, Inject } from '@syncfusion/ej2-
react-charts';
import { synchronizedData } from 'datasource.ts';
import * as React from 'react';
import { Browser } from '@syncfusion/ej2-base';
function App() {
    let chart1 = React.useRef(null);
    let chart2 = React.useRef(null);
    let charts = [];
    React.useEffect(() => {
        charts = [chart1.current, chart2.current];
    }, []);
    let zoomFactor = 0;
    let zoomPosition = 0;
    let zoomComplete = (args) => {
        if (args.axis.name === 'primaryXAxis') {
            zoomFactor = args.currentZoomFactor;
            zoomPosition = args.currentZoomPosition;
            zoomCompleteFunction(args);
        }
    };
    let zoomCompleteFunction = (args) => {
        for (let i = 0; i < charts.length; i++) {
            if (args.axis.series[0].chart.element.id !==
charts[i].element.id) {
                charts[i].primaryXAxis.zoomFactor = zoomFactor;
                charts[i].primaryXAxis.zoomPosition = zoomPosition;
                charts[i].zoomModule.isZoomed =
args.axis.series[0].chart.zoomModule.isZoomed;
                charts[i].zoomModule.isPanning =
args.axis.series[0].chart.zoomModule.isPanning;
            }
        }
    }
    return <div className="control-section">
        <div className="row">
            <div className="col">
                <ChartComponent
                    id="container1"
                    ref={chart1}
                    primaryXAxis={ {
```

```

        minimum: new Date(2023, 1, 18),
        maximum: new Date(2023, 7, 18),
        valueType: 'DateTime',
        labelFormat: 'MMM d',
        lineStyle: { width: 0 },
        majorGridLines: { width: 0 },
        edgeLabelPlacement: Browser.isDevice ? 'None' :
'Shift',

        labelRotation: Browser.isDevice ? -45 : 0,
        interval: Browser.isDevice ? 2 : 1
    }}
    primaryYAxis={{
        labelFormat: 'n2',
        majorTickLines: { width: 0 },
        lineStyle: { width: 0 },
        minimum: 0.86,
        maximum: 0.96,
        interval: 0.025
    }}
    chartArea={{ border: { width: 0 } }}
    zoomSettings={{
        enableMouseWheelZooming: true,
        enablePinchZooming: true,
        enableScrollbar: false,
        enableDeferredZooming: false,
        enableSelectionZooming: true,
        enablePan: true,
        mode: 'X',
        toolbarItems: ['Pan', 'Reset']
    }}
    zoomComplete={zoomComplete.bind(this)}
    titleStyle={{ textAlign: 'Near' }}
    title="US to Euro"
    <Inject services={[LineSeries, DateTime, Zoom,
Selection]} />

    <SeriesCollectionDirective>
        <SeriesDirective type="Line"
dataSource={synchronizedData} xName="USD" yName="EUR" width={2}
emptyPointSettings={{ mode: 'Drop' }}></SeriesDirective>
    </SeriesCollectionDirective>
    </ChartComponent>
</div>
<div className="col">
    <ChartComponent
        id="container2"
        ref={chart2}
        primaryXAxis={{
            minimum: new Date(2023, 1, 18),
            maximum: new Date(2023, 7, 18),
            valueType: 'DateTime',
            labelFormat: 'MMM d',
            lineStyle: { width: 0 },
            majorGridLines: { width: 0 },
            edgeLabelPlacement: Browser.isDevice ? 'None' :
'Shift',

            labelRotation: Browser.isDevice ? -45 : 0,
            interval: Browser.isDevice ? 2 : 1
        }}
    />
</div>

```

```

    }}
    primaryYAxis={{
      labelFormat: 'n1',
      majorTickLines: { width: 0 },
      lineStyle: { width: 0 },
      minimum: 79,
      maximum: 85,
      interval: 1.5
    }}
    chartArea={{ border: { width: 0 } }}
    zoomSettings={{
      enableMouseWheelZooming: true,
      enablePinchZooming: true,
      enableScrollbar: false,
      enableDeferredZooming: false,
      enableSelectionZooming: true,
      enablePan: true,
      mode: 'X',
      toolbarItems: ['Pan', 'Reset']
    }}
    zoomComplete={zoomComplete.bind(this)}
    titleStyle={{ textAlign: 'Near' }}
    title="US to INR">
    <Inject services={[SplineAreaSeries, DateTime, Zoom,
Selection]} />

    <SeriesCollectionDirective>
      <SeriesDirective type="SplineArea"
dataSource={synchronizedData} xName="USD" yName="INR" opacity={0.6}
width={2} border={{ width: 2 }}></SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
</div>
</div>
</div>
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as ReactDOM from "react-dom";
import { Chart, SplineAreaSeries, LineSeries, DateTime, Zoom,
IZoomCompleteEventArgs, Selection, ChartComponent,
SeriesCollectionDirective, SeriesDirective, Inject } from '@syncfusion/ej2-
react-charts';
import { synchronizedData } from 'datasource.ts';
import * as React from "react";
import { Browser } from '@syncfusion/ej2-base';
function App() {
  let chart1 = React.useRef<ChartComponent>(null);
  let chart2 = React.useRef<ChartComponent>(null);
  let charts: ChartComponent[] = [];
  React.useEffect(() => {

```



```

        charts = [chart1.current, chart2.current];
    }, []);
    let zoomFactor: number = 0;
    let zoomPosition: number = 0;
    let zoomComplete = (args: IZoomCompleteEventArgs): void => {
        if (args.axis.name === 'primaryXAxis') {
            zoomFactor = args.currentZoomFactor;
            zoomPosition = args.currentZoomPosition;
            zoomCompleteFunction(args);
        }
    };
    let zoomCompleteFunction = (args: IZoomCompleteEventArgs): void => {
        for (let i: number = 0; i < charts.length; i++) {
            if (args.axis.series[0].chart.element.id !==
charts[i].element.id) {
                charts[i].primaryXAxis.zoomFactor = zoomFactor;
                charts[i].primaryXAxis.zoomPosition = zoomPosition;
                charts[i].zoomModule.isZoomed =
args.axis.series[0].chart.zoomModule.isZoomed;
                charts[i].zoomModule.isPanning =
args.axis.series[0].chart.zoomModule.isPanning;
            }
        }
    }
    return <div className="control-section">
        <div className="row">
            <div className="col">
                <ChartComponent
                    id="container1"
                    ref={chart1}
                    primaryXAxis={{
                        minimum: new Date(2023, 1, 18),
                        maximum: new Date(2023, 7, 18),
                        valueType: 'DateTime',
                        labelFormat: 'MMM d',
                        lineStyle: { width: 0 },
                        majorGridLines: { width: 0 },
                        edgeLabelPlacement: Browser.isDevice ? 'None' :
'Shift',

                        labelRotation: Browser.isDevice ? -45 : 0,
                        interval: Browser.isDevice ? 2 : 1
                    }}
                    primaryYAxis={{
                        labelFormat: 'n2',
                        majorTickLines: { width: 0 },
                        lineStyle: { width: 0 },
                        minimum: 0.86,
                        maximum: 0.96,
                        interval: 0.025
                    }}
                    chartArea={{ border: { width: 0 } }}
                    zoomSettings={{
                        enableMouseWheelZooming: true,
                        enablePinchZooming: true,
                        enableScrollbar: false,
                        enableDeferredZooming: false,
                        enableSelectionZooming: true,

```

```

        enablePan: true,
        mode: 'X',
        toolbarItems: ['Pan', 'Reset']
    })
    zoomComplete={zoomComplete.bind(this)}
    titleStyle={{ textAlign: 'Near' }}
    title="US to Euro">
    <Inject services={[LineSeries, DateTime, Zoom,
Selection]] />

    <SeriesCollectionDirective>
        <SeriesDirective type="Line"
dataSource={synchronizedData} xName="USD" yName="EUR" width={2}
emptyPointSettings={{ mode: 'Drop' }}></SeriesDirective>
    </SeriesCollectionDirective>
    </ChartComponent>
</div>
<div className="col">
    <ChartComponent
        id="container2"
        ref={chart2}
        primaryXAxis={{
            minimum: new Date(2023, 1, 18),
            maximum: new Date(2023, 7, 18),
            valueType: 'DateTime',
            labelFormat: 'MMM d',
            lineStyle: { width: 0 },
            majorGridLines: { width: 0 },
            edgeLabelPlacement: Browser.isDevice ? 'None' :
'Shift',

            labelRotation: Browser.isDevice ? -45 : 0,
            interval: Browser.isDevice ? 2 : 1
        }}
        primaryYAxis={{
            labelFormat: 'n1',
            majorTickLines: { width: 0 },
            lineStyle: { width: 0 },
            minimum: 79,
            maximum: 85,
            interval: 1.5
        }}
        chartArea={{ border: { width: 0 } }}
        zoomSettings={{
            enableMouseWheelZooming: true,
            enablePinchZooming: true,
            enableScrollbar: false,
            enableDeferredZooming: false,
            enableSelectionZooming: true,
            enablePan: true,
            mode: 'X',
            toolbarItems: ['Pan', 'Reset']
        }}
        zoomComplete={zoomComplete.bind(this)}
        titleStyle={{ textAlign: 'Near' }}
        title="US to INR">
    <Inject services={[SplineAreaSeries, DateTime, Zoom,
Selection]] />

    <SeriesCollectionDirective>

```

```

        <SeriesDirective type="SplineArea"
dataSource={synchronizedData} xName="USD" yName="INR" opacity={0.6}
width={2} border={{ width: 2 }}></SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
</div>
</div>
</div>
</div>
</div>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Selection synchronization

You can select the data across multiple charts using the [selectionComplete](#) event. In the [selectionComplete](#) event, obtain the selected values of the particular chart, and then apply those values to the other charts.

INDEX.JSX

```

{% raw %}
import * as ReactDOM from "react-dom";
import { Chart, SplineSeries, LineSeries, DateTime, Zoom,
IZoomCompleteEventArgs, Selection, ISelectionCompleteEventArgs,
ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject } from
'@syncfusion/ej2-react-charts';
import { synchronizedData } from 'datasource.ts';
import * as React from 'react';
import { Browser } from '@syncfusion/ej2-base';
function App() {
    let chart1 = React.useRef(null);
    let chart2 = React.useRef(null);
    let charts = [];
    React.useEffect(() => {
        charts = [chart1.current, chart2.current];
    }, []);
    let zoomFactor = 0;
    let zoomPosition = 0;
    let count = 0;
    let zoomComplete = (args) => {
        if (args.axis.name === 'primaryXAxis') {
            zoomFactor = args.currentZoomFactor;
            zoomPosition = args.currentZoomPosition;
            zoomCompleteFunction(args);
        }
    };
    let zoomCompleteFunction = (args) => {
        for (let i = 0; i < charts.length; i++) {
            if (args.axis.series[0].chart.element.id !==
charts[i].element.id) {
                charts[i].primaryXAxis.zoomFactor = zoomFactor;
                charts[i].primaryXAxis.zoomPosition = zoomPosition;
                charts[i].zoomModule.isZoomed =
args.axis.series[0].chart.zoomModule.isZoomed;
                charts[i].zoomModule.isPanning =
args.axis.series[0].chart.zoomModule.isPanning;
            }
        }
    };
}

```

```

    }
  }
};
let selectionComplete = (args) => {
  selectionCompleteFunction(args);
};
let selectionCompleteFunction = (args) => {
  if (count == 0) {
    for (var j = 0; j < args.selectedDataValues.length; j++) {
      args.selectedDataValues[j].point =
args.selectedDataValues[j].pointIndex;
      args.selectedDataValues[j].series =
args.selectedDataValues[j].seriesIndex;
    }
    for (var i = 0; i < charts.length; i++) {
      if (args.chart.element.id !== charts[i].element.id) {
        charts[i].selectedDataIndexes = args.selectedDataValues;
        count += 1;
        charts[i].dataBind();
      }
    }
    count = 0;
  }
};
return <div className="control-section">
  <div className="row">
    <div className="col">
      <ChartComponent
        id="container1"
        ref={chart1}
        primaryXAxis={{
          minimum: new Date(2023, 1, 18),
          maximum: new Date(2023, 7, 18),
          valueType: 'DateTime',
          labelFormat: 'MMM d',
          lineStyle: { width: 0 },
          majorGridLines: { width: 0 },
          edgeLabelPlacement: Browser.isDevice ? 'None' :
'Shift',

          labelRotation: Browser.isDevice ? -45 : 0,
          interval: Browser.isDevice ? 2 : 1
        }}
        primaryYAxis={{
          labelFormat: 'n2',
          majorTickLines: { width: 0 },
          lineStyle: { width: 0 },
          minimum: 0.86,
          maximum: 0.96,
          interval: 0.025
        }}
        chartArea={{ border: { width: 0 } }}
        zoomSettings={{
          enableSelectionZooming: true,
          mode: 'X'
        }}
        zoomComplete={zoomComplete.bind(this)}
        selectionComplete={selectionComplete.bind(this)}

```

```

        titleStyle={{ textAlign: 'Near' }}
        title="US to Euro"
        selectionMode='Point'
        selectionPattern='Box'>
        <Inject services={[LineSeries, DateTime, Zoom,
Selection]] />

        <SeriesCollectionDirective>
            <SeriesDirective type="Line"
dataSource={synchronizedData} xName="USD" yName="EUR" width={2}
emptyPointSettings={{ mode: 'Drop' }}></SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
</div>
<div className="col">
    <ChartComponent
        id="container2"
        ref={chart2}
        primaryXAxis={{
            minimum: new Date(2023, 1, 18),
            maximum: new Date(2023, 7, 18),
            valueType: 'DateTime',
            labelFormat: 'MMM d',
            lineStyle: { width: 0 },
            majorGridLines: { width: 0 },
            edgeLabelPlacement: Browser.isDevice ? 'None' :
'Shift',

            labelRotation: Browser.isDevice ? -45 : 0,
            interval: Browser.isDevice ? 2 : 1
        }}
        primaryYAxis={{
            labelFormat: 'n1',
            majorTickLines: { width: 0 },
            lineStyle: { width: 0 },
            minimum: 79,
            maximum: 85,
            interval: 1.5
        }}
        chartArea={{ border: { width: 0 } }}
        zoomSettings={{
            enableSelectionZooming: true,
            mode: 'X'
        }}
        zoomComplete={zoomComplete.bind(this)}
        selectionComplete={selectionComplete.bind(this)}
        titleStyle={{ textAlign: 'Near' }}
        title="US to INR"
        selectionMode='Point'
        selectionPattern='Box'>
        <Inject services={[SplineSeries, DateTime, Zoom,
Selection]] />

        <SeriesCollectionDirective>
            <SeriesDirective type="Spline"
dataSource={synchronizedData} xName="USD" yName="INR" width={2} border={{
width: 2 }}></SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
</div>

```

```

        </div>
    </div>
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as ReactDOM from "react-dom";
import { Chart, SplineSeries, LineSeries, DateTime, Zoom,
IZoomCompleteEventArgs, Selection, ISelectionCompleteEventArgs,
ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject } from
'@syncfusion/ej2-react-charts';
import { synchronizedData } from 'datasource.ts';
import * as React from "react";
import { Browser } from '@syncfusion/ej2-base';
function App() {
    let chart1 = React.useRef<ChartComponent>(null);
    let chart2 = React.useRef<ChartComponent>(null);
    let charts: ChartComponent[] = [];
    React.useEffect(() => {
        charts = [chart1.current, chart2.current];
    }, []);
    let zoomFactor: number = 0;
    let zoomPosition: number = 0;
    let count: number = 0;
    let zoomComplete = (args: IZoomCompleteEventArgs): void => {
        if (args.axis.name === 'primaryXAxis') {
            zoomFactor = args.currentZoomFactor;
            zoomPosition = args.currentZoomPosition;
            zoomCompleteFunction(args);
        }
    };
    let zoomCompleteFunction = (args: IZoomCompleteEventArgs): void => {
        for (let i: number = 0; i < charts.length; i++) {
            if (args.axis.series[0].chart.element.id !==
charts[i].element.id) {
                charts[i].primaryXAxis.zoomFactor = zoomFactor;
                charts[i].primaryXAxis.zoomPosition = zoomPosition;
                charts[i].zoomModule.isZoomed =
args.axis.series[0].chart.zoomModule.isZoomed;
                charts[i].zoomModule.isPanning =
args.axis.series[0].chart.zoomModule.isPanning;
            }
        }
    };
    let selectionComplete = (args: ISelectionCompleteEventArgs): void => {
        selectionCompleteFunction(args);
    };
    let selectionCompleteFunction = (args: ISelectionCompleteEventArgs):
void => {
        if (count == 0) {
            for (var j = 0; j < args.selectedDataValues.length; j++) {

```

```

        args.selectedDataValues[j].point =
args.selectedDataValues[j].pointIndex;
        args.selectedDataValues[j].series =
args.selectedDataValues[j].seriesIndex;
    }
    for (var i = 0; i < charts.length; i++) {
        if (args.chart.element.id !== charts[i].element.id) {
            charts[i].selectedDataIndexes = args.selectedDataValues;
            count += 1;
            charts[i].dataBind();
        }
    }
    count = 0;
}
};
return <div className="control-section">
    <div className="row">
        <div className="col">
            <ChartComponent
                id="container1"
                ref={chart1}
                primaryXAxis={{
                    minimum: new Date(2023, 1, 18),
                    maximum: new Date(2023, 7, 18),
                    valueType: 'DateTime',
                    labelFormat: 'MMM d',
                    lineStyle: { width: 0 },
                    majorGridLines: { width: 0 },
                    edgeLabelPlacement: Browser.isDevice ? 'None' :
'Shift',

                    labelRotation: Browser.isDevice ? -45 : 0,
                    interval: Browser.isDevice ? 2 : 1
                }}
                primaryYAxis={{
                    labelFormat: 'n2',
                    majorTickLines: { width: 0 },
                    lineStyle: { width: 0 },
                    minimum: 0.86,
                    maximum: 0.96,
                    interval: 0.025
                }}
                chartArea={{ border: { width: 0 } }}
                zoomSettings={{
                    enableSelectionZooming: true,
                    mode: 'X'
                }}
                zoomComplete={zoomComplete.bind(this)}
                selectionComplete={selectionComplete.bind(this)}
                titleStyle={{ textAlign: 'Near' }}
                title="US to Euro"
                selectionMode='Point'
                selectionPattern='Box'>
                <Inject services={[LineSeries, DateTime, Zoom,
Selection]} />
                <SeriesCollectionDirective>

```

```

        <SeriesDirective type="Line"
dataSource={synchronizedData} xName="USD" yName="EUR" width={2}
emptyPointSettings={{ mode: 'Drop' }}></SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
</div>
<div className="col">
    <ChartComponent
        id="container2"
        ref={chart2}
        primaryXAxis={{
            minimum: new Date(2023, 1, 18),
            maximum: new Date(2023, 7, 18),
            valueType: 'DateTime',
            labelFormat: 'MMM d',
            lineStyle: { width: 0 },
            majorGridLines: { width: 0 },
            edgeLabelPlacement: Browser.isDevice ? 'None' :
'Shift',

            labelRotation: Browser.isDevice ? -45 : 0,
            interval: Browser.isDevice ? 2 : 1
        }}
        primaryYAxis={{
            labelFormat: 'n1',
            majorTickLines: { width: 0 },
            lineStyle: { width: 0 },
            minimum: 79,
            maximum: 85,
            interval: 1.5
        }}
        chartArea={{ border: { width: 0 } }}
        zoomSettings={{
            enableSelectionZooming: true,
            mode: 'X'
        }}
        zoomComplete={zoomComplete.bind(this)}
        selectionComplete={selectionComplete.bind(this)}
        titleStyle={{ textAlign: 'Near' }}
        title="US to INR"
        selectionMode='Point'
        selectionPattern='Box'>
        <Inject services={[SplineSeries, DateTime, Zoom,
Selection]} />
        <SeriesCollectionDirective>
            <SeriesDirective type="Spline"
dataSource={synchronizedData} xName="USD" yName="INR" width={2} border={{
width: 2 }}></SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
</div>
</div>
</div>
</div>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```


<!-- markdownlint-disable MD036 -->

Selection in React Chart component

Chart provides selection support for the series and its data points on mouse click.

When Mouse is clicked on the data points, the corresponding series legend will also be selected.

We have different type of selection mode for selecting the data. They are,

- None
- Point
- Series
- Cluster
- DragXY
- DragX
- DragY

Point

You can select a point, by setting `selectionMode` to point.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries, Selection } from
"@syncfusion/ej2-react-charts";
import { selectionData } from "datasource.ts";
function App() {
    const primaryXAxis = { valueType: 'Category' };
    return <ChartComponent id='charts' primaryXAxis={primaryXAxis}
title='Olympic Medals' selectionMode='Point'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Selection, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='gold' name='Gold' type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='silver' name='Silver' type='Column'></SeriesDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='bronze' name='Bronze' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
```

```
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
      Legend, Category, Tooltip, DataLabel, ColumnSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
  const primaryxAxis: AxisModel = { valueType: 'Category' };
  const selectionData: any[] = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    title='Olympic Medals' selectionMode='Point'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Selection, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={selectionData} xName='country'
yName='gold' name='Gold' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={selectionData} xName='country'
yName='silver' name='Silver' type='Column'></SeriesDirective>
      <SeriesDirective dataSource={selectionData} xName='country'
yName='bronze' name='Bronze' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Series

You can select a series, by setting `selectionMode` to series.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries, Selection } from
 '@syncfusion/ej2-react-charts';
import { selectionData } from 'datasource.ts';
function App() {
  const primaryxAxis = { valueType: 'Category' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
title='Olympic Medals' selectionMode='Series'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Selection, Category]} />
    <SeriesCollectionDirective>
```

```

        <SeriesDirective dataSource={selectionData} xName='country'
yName='gold' name='Gold' type='Column'>
        </SeriesDirective>
        <SeriesDirective dataSource={selectionData} xName='country'
yName='silver' name='Silver' type='Column'>
        </SeriesDirective>
        <SeriesDirective dataSource={selectionData} xName='country'
yName='bronze' name='Bronze' type='Column'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    Legend, Category, Tooltip, DataLabel, ColumnSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Category' };
    const selectionData: any[] = [
        { country: "USA", gold: 50, silver: 70, bronze: 45 },
        { country: "China", gold: 40, silver: 60, bronze: 55 },
        { country: "Japan", gold: 70, silver: 60, bronze: 50 },
        { country: "Australia", gold: 60, silver: 56, bronze: 40 },
        { country: "France", gold: 50, silver: 45, bronze: 35 },
        { country: "Germany", gold: 40, silver: 30, bronze: 22 },
        { country: "Italy", gold: 40, silver: 35, bronze: 37 },
        { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        title='Olympic Medals' selectionMode='Series'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Selection, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='gold' name='Gold' type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='silver' name='Silver' type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='bronze' name='Bronze' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Cluster

You can select the points that corresponds to the same index in all the series, by setting `selectionMode` to cluster.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries, Selection } from
"@syncfusion/ej2-react-charts";
import { selectionData } from "datasource.ts";
function App() {
    const primaryxAxis = { valueType: 'Category' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
title='Olympic Medals' selectionMode='Cluster'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Selection, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='gold' name='Gold' type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='silver' name='Silver' type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='bronze' name='Bronze' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries, Selection}
from "@syncfusion/ej2-react-charts";
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Category' };
    const selectionData: any[] = [
        { country: "USA", gold: 50, silver: 70, bronze: 45 },
        { country: "China", gold: 40, silver: 60, bronze: 55 },
        { country: "Japan", gold: 70, silver: 60, bronze: 50 },
        { country: "Australia", gold: 60, silver: 56, bronze: 40 },
        { country: "France", gold: 50, silver: 45, bronze: 35 },
        { country: "Germany", gold: 40, silver: 30, bronze: 22 },
        { country: "Italy", gold: 40, silver: 35, bronze: 37 },
        { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
    ]
```

```

];
return <ChartComponent id='charts'
  primaryXAxis={primaryXAxis}
  title='Olympic Medals' selectionMode='Cluster'>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Selection, Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={selectionData} xName='country'
yName='gold' name='Gold' type='Column'>
    </SeriesDirective>
    <SeriesDirective dataSource={selectionData} xName='country'
yName='silver' name='Silver' type='Column'>
    </SeriesDirective>
    <SeriesDirective dataSource={selectionData} xName='country'
yName='bronze' name='Bronze' type='Column'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Rectangular selection

DragXY, DragX and DragY

To fetch the collection of data under a particular region, you have to set `selectionMode` as `DragXY`.

- DragXY - Allows us to select data with respect to horizontal and vertical axis.
- DragX - Allows us to select data with respect to horizontal axis.
- DragY - Allows us to select data with respect to vertical axis.

The selected data's are returned as an array collection in the [dragComplete](#) event.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Tooltip, Selection, ScatterSeries } from '@syncfusion/ej2-react-
charts';
import { dragData } from 'datasource.ts';
function App() {
  const marker = { width: 12, height: 12 };
  return <ChartComponent id='charts' selectionMode='DragXY' title='Height
Vs Weight'>
    <Inject services={[ScatterSeries, Legend, Tooltip, Selection]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={dragData} xName='x' yName='y'
type='Scatter' marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;

```

```
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
    Legend, Category, Tooltip, Selection, DataLabel, ScatterSeries }
from '@syncfusion/ej2-react-charts';
function App() {
    const marker = { width: 12, height: 12 };
    const dragData: any[] = [
        { x: 1971, y: 50 }, { x: 1972, y: 20 }, { x: 1973, y: 63 }, { x: 1974,
y: 81 }, { x: 1975, y: 64 },
        { x: 1976, y: 36 }, { x: 1977, y: 22 }, { x: 1978, y: 78 }, { x: 1979,
y: 60 }, { x: 1980, y: 41 },
        { x: 1981, y: 62 }, { x: 1982, y: 56 }, { x: 1983, y: 96 }, { x: 1984,
y: 48 }, { x: 1985, y: 23 },
        { x: 1986, y: 54 }, { x: 1987, y: 73 }, { x: 1988, y: 56 }, { x: 1989,
y: 67 }, { x: 1990, y: 79 },
        { x: 1991, y: 18 }, { x: 1992, y: 78 }, { x: 1993, y: 92 }, { x: 1994,
y: 43 }, { x: 1995, y: 29 },
        { x: 1996, y: 14 }, { x: 1997, y: 85 }, { x: 1998, y: 24 }, { x: 1999,
y: 61 }, { x: 2000, y: 80 },
        { x: 2001, y: 14 }, { x: 2002, y: 34 }, { x: 2003, y: 81 }, { x: 2004,
y: 70 }, { x: 2005, y: 21 },
        { x: 2006, y: 70 }, { x: 2007, y: 32 }, { x: 2008, y: 43 }, { x: 2009,
y: 21 }, { x: 2010, y: 63 },
        { x: 2011, y: 9 }, { x: 2012, y: 51 }, { x: 2013, y: 25 }, { x: 2014, y:
96 }, { x: 2015, y: 32 }
    ];
    return <ChartComponent id='charts'
        selectionMode='DragXY'
        title='Height Vs Weight'>
        <Inject services={[ScatterSeries, Legend, Tooltip, Selection]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={dragData} xName='x' yName='y'
type='Scatter'
                marker={marker}>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>
    </>;
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));
```

Lasso selection

To select a region by drawing freehand shapes to fetch a collection of data use `selectionMode` as `Lasso`. You can also select multiple regions on the chart through this mode.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
```

```
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Tooltip, Selection, ScatterSeries } from '@syncfusion/ej2-react-
charts';
import { dragData } from 'datasource.ts';
function App() {
    const marker = { width: 12, height: 12 };
    return <ChartComponent id='charts' selectionMode='Lasso' title='Height
Vs Weight'>
        <Inject services={[ScatterSeries, Legend, Tooltip, Selection]}>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={dragData} xName='x' yName='y'
type='Scatter' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, Selection, DataLabel, ScatterSeries }
from '@syncfusion/ej2-react-charts';
function App() {
    const marker = { width: 12, height: 12 };
    const dragData: any[] = [
        { x: 1971, y: 50 }, { x: 1972, y: 20 }, { x: 1973, y: 63 }, { x: 1974,
y: 81 }, { x: 1975, y: 64 },
        { x: 1976, y: 36 }, { x: 1977, y: 22 }, { x: 1978, y: 78 }, { x: 1979,
y: 60 }, { x: 1980, y: 41 },
        { x: 1981, y: 62 }, { x: 1982, y: 56 }, { x: 1983, y: 96 }, { x: 1984,
y: 48 }, { x: 1985, y: 23 },
        { x: 1986, y: 54 }, { x: 1987, y: 73 }, { x: 1988, y: 56 }, { x: 1989,
y: 67 }, { x: 1990, y: 79 },
        { x: 1991, y: 18 }, { x: 1992, y: 78 }, { x: 1993, y: 92 }, { x: 1994,
y: 43 }, { x: 1995, y: 29 },
        { x: 1996, y: 14 }, { x: 1997, y: 85 }, { x: 1998, y: 24 }, { x: 1999,
y: 61 }, { x: 2000, y: 80 },
        { x: 2001, y: 14 }, { x: 2002, y: 34 }, { x: 2003, y: 81 }, { x: 2004,
y: 70 }, { x: 2005, y: 21 },
        { x: 2006, y: 70 }, { x: 2007, y: 32 }, { x: 2008, y: 43 }, { x: 2009,
y: 21 }, { x: 2010, y: 63 },
        { x: 2011, y: 9 }, { x: 2012, y: 51 }, { x: 2013, y: 25 }, { x: 2014, y:
96 }, { x: 2015, y: 32 }
    ];
    return <ChartComponent id='charts'
        selectionMode='Lasso'
        title='Height Vs Weight'>
        <Inject services={[ScatterSeries, Legend, Tooltip, Selection]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={dragData} xName='x' yName='y'
type='Scatter'>
```

```

        marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Multi-region selection

To select multiple region on the chart, set the `allowMultiSelection` property to true.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Tooltip, Selection, ScatterSeries } from '@syncfusion/ej2-react-
charts';
import { dragData } from 'datasource.ts';
function App() {
  const marker = { width: 12, height: 12 };
  return <ChartComponent id='charts' selectionMode='Lasso'
allowMultiSelection={true} title='Height Vs Weight'>
    <Inject services={[ScatterSeries, Legend, Tooltip, Selection]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={dragData} xName='x' yName='y'
type='Scatter' marker={marker}>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, Selection, DataLabel, ScatterSeries}
from '@syncfusion/ej2-react-charts';
function App() {
  const marker = { width: 12, height: 12 };
  const dragData: any[] = [
    { x: 1971, y: 50 }, { x: 1972, y: 20 }, { x: 1973, y: 63 }, { x: 1974,
y: 81 }, { x: 1975, y: 64 },
    { x: 1976, y: 36 }, { x: 1977, y: 22 }, { x: 1978, y: 78 }, { x: 1979,
y: 60 }, { x: 1980, y: 41 },
    { x: 1981, y: 62 }, { x: 1982, y: 56 }, { x: 1983, y: 96 }, { x: 1984,
y: 48 }, { x: 1985, y: 23 },
    { x: 1986, y: 54 }, { x: 1987, y: 73 }, { x: 1988, y: 56 }, { x: 1989,
y: 67 }, { x: 1990, y: 79 },
    { x: 1991, y: 18 }, { x: 1992, y: 78 }, { x: 1993, y: 92 }, { x: 1994,
y: 43 }, { x: 1995, y: 29 },

```



```

    { x: 1996, y: 14 }, { x: 1997, y: 85 }, { x: 1998, y: 24 }, { x: 1999,
y: 61 }, { x: 2000, y: 80 },
    { x: 2001, y: 14 }, { x: 2002, y: 34 }, { x: 2003, y: 81 }, { x: 2004,
y: 70 }, { x: 2005, y: 21 },
    { x: 2006, y: 70 }, { x: 2007, y: 32 }, { x: 2008, y: 43 }, { x: 2009,
y: 21 }, { x: 2010, y: 63 },
    { x: 2011, y: 9 }, { x: 2012, y: 51 }, { x: 2013, y: 25 }, { x: 2014, y:
96 }, { x: 2015, y: 32 }
  ];
  return <ChartComponent id='charts'
    selectionMode='Lasso' allowMultiSelection={true}
    title='Height Vs Weight'>
    <Inject services={[ScatterSeries, Legend, Tooltip, Selection]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={dragData} xName='x' yName='y'
type='Scatter'
        marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Selection Type

You can select multiple points or series, by enabling the [isMultiSelect](#) property.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries, Selection } from
'@syncfusion/ej2-react-charts';
import { selectionData } from 'datasource.ts';
function App() {
  const primaryXAxis = { valueType: 'Category' };
  return <ChartComponent id='charts' primaryXAxis={primaryXAxis}
title='Olympic Medals' isMultiSelect={true} selectionMode='Point'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Selection, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={selectionData} xName='country'
yName='gold' name='Gold' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={selectionData} xName='country'
yName='silver' name='Silver' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={selectionData} xName='country'
yName='bronze' name='Bronze' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;

```

```
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    Legend, Category, Tooltip, DataLabel, ColumnSeries, Selection }
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryXAxis: AxisModel = { valueType: 'Category' };
    const selectionData: any[] = [
        { country: "USA", gold: 50, silver: 70, bronze: 45 },
        { country: "China", gold: 40, silver: 60, bronze: 55 },
        { country: "Japan", gold: 70, silver: 60, bronze: 50 },
        { country: "Australia", gold: 60, silver: 56, bronze: 40 },
        { country: "France", gold: 50, silver: 45, bronze: 35 },
        { country: "Germany", gold: 40, silver: 30, bronze: 22 },
        { country: "Italy", gold: 40, silver: 35, bronze: 37 },
        { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryXAxis}
        title='Olympic Medals' isMultiSelect={true} selectionMode='Point'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Selection, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='gold' name='Gold' type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='silver' name='Silver' type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='bronze' name='Bronze' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Selection on Load

You can able to select a point or series programmatically on a chart using [selectedDataIndexes](#) property.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries, Selection } from
 '@syncfusion/ej2-react-charts';
import { selectionData } from 'datasource.ts';
function App() {
    const selectedData = [{ series: 0, point: 1 }, { series: 2, point: 3 }];
```

```

const primaryxAxis = { valueType: 'Category' };
const animation = { enable: false };
return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
title='Olympic Medals' selectedDataIndexes={selectedData}
isMultiSelect={true} selectionMode='Point'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Selection, Category]}>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='gold' name='Gold' type='Column' selectionStyle='chartSelection1'
animation={animation}></SeriesDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='silver' name='Silver' type='Column' selectionStyle='chartSelection2'
animation={animation}></SeriesDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='bronze' name='Bronze' type='Column' selectionStyle='chartSelection3'
animation={animation}></SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
    Legend, Category, Tooltip, DataLabel, ColumnSeries, Selection }
from '@syncfusion/ej2-react-charts';
function App() {
    const selectedData: any[] = [{ series: 0, point: 1 }, { series: 2, point:
3 }];
    const primaryxAxis: AxisModel = { valueType: 'Category' };
    const animation = { enable: false };
    const selectionData: any[] = [
        { country: "USA", gold: 50, silver: 70, bronze: 45 },
        { country: "China", gold: 40, silver: 60, bronze: 55 },
        { country: "Japan", gold: 70, silver: 60, bronze: 50 },
        { country: "Australia", gold: 60, silver: 56, bronze: 40 },
        { country: "France", gold: 50, silver: 45, bronze: 35 },
        { country: "Germany", gold: 40, silver: 30, bronze: 22 },
        { country: "Italy", gold: 40, silver: 35, bronze: 37 },
        { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
    ];
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        title='Olympic Medals' selectedDataIndexes={selectedData}
        isMultiSelect={true} selectionMode='Point'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Selection, Category]} />
        <SeriesCollectionDirective>

```

```

        <SeriesDirective dataSource={selectionData} xName='country'
yName='gold' name='Gold' type='Column' selectionStyle='chartSelection1'
animation={animation} ></SeriesDirective>
        <SeriesDirective dataSource={selectionData} xName='country'
yName='silver' name='Silver' type='Column' selectionStyle='chartSelection2'
animation={animation}></SeriesDirective>
        <SeriesDirective dataSource={selectionData} xName='country'
yName='bronze' name='Bronze' type='Column' selectionStyle='chartSelection3'
animation={animation}></SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Selection through on legend

You can able to select a point or series through on legend using [toggleVisibility](#) property. Also, use [enableHighlight](#) property for highlighting the series through legend.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries, Highlight, Selection }
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis = { valueType: 'Category' };
    const legendSettings = { visible: true, toggleVisibility: false,
enableHighlight: true };
    const selectionData = [
        { country: "USA", gold: 50, silver: 70, bronze: 45 },
        { country: "China", gold: 40, silver: 60, bronze: 55 },
        { country: "Japan", gold: 70, silver: 60, bronze: 50 },
        { country: "Australia", gold: 60, silver: 56, bronze: 40 },
        { country: "France", gold: 50, silver: 45, bronze: 35 },
        { country: "Germany", gold: 40, silver: 30, bronze: 22 },
        { country: "Italy", gold: 40, silver: 35, bronze: 37 },
        { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
    ];
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
legendSettings={legendSettings} title='Olympic Medals'
selectionMode='Cluster'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Selection,
Category, Highlight]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='gold' name='Gold' type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='silver' name='Silver' type='Column'>
            </SeriesDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='bronze' name='Bronze' type='Column'>
            </SeriesDirective>

```

```

    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
  AxisModel, ChartComponent, SeriesCollectionDirective, SeriesDirective,
  Inject,
  Legend, Category, Tooltip, DataLabel, ColumnSeries, Highlight,
  LegendSettingsModel, Selection
}
  from '@syncfusion/ej2-react-charts';
function App() {
  const primaryXAxis: AxisModel = { valueType: 'Category' };
  const legendSettings: LegendSettingsModel = { visible: true,
toggleVisibility: false, enableHighlight: true };
  const selectionData: any[] = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryXAxis}
    title='Olympic Medals' selectionMode='Cluster'
    legendSettings={legendSettings}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Selection,
Category, Highlight]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={selectionData} xName='country'
yName='gold' name='Gold' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={selectionData} xName='country'
yName='silver' name='Silver' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={selectionData} xName='country'
yName='bronze' name='Bronze' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Customization for selection

You can apply custom style to selected points or series with [selectionStyle](#) property.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries, Selection } from
'@syncfusion/ej2-react-charts';
import { selectionData } from 'datasource.ts';
function App() {
    const primaryxAxis = { valueType: 'Category', title: 'Countries' };
    const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Olympic Medals' isMultiSelect={true}
selectionMode='Point'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Selection, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='gold' name='Gold' type='Column'
selectionStyle='chartSelection1'></SeriesDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='silver' name='Silver' type='Column'
selectionStyle='chartSelection2'></SeriesDirective>
            <SeriesDirective dataSource={selectionData} xName='country'
yName='bronze' name='Bronze' type='Column'
selectionStyle='chartSelection3'></SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries, Selection }
from '@syncfusion/ej2-react-charts';
function App() {
    const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
    const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
    const selectionData: any[] = [
        { country: "USA", gold: 50, silver: 70, bronze: 45 },
        { country: "China", gold: 40, silver: 60, bronze: 55 },
        { country: "Japan", gold: 70, silver: 60, bronze: 50 },
        { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    ];
```

```

    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Olympic Medals' isMultiSelect={true} selectionMode='Point'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Selection, Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={selectionData} xName='country'
yName='gold' name='Gold' type='Column'
selectionStyle='chartSelection1'></SeriesDirective>
      <SeriesDirective dataSource={selectionData} xName='country'
yName='silver' name='Silver' type='Column'
selectionStyle='chartSelection2'></SeriesDirective>
      <SeriesDirective dataSource={selectionData} xName='country'
yName='bronze' name='Bronze' type='Column'
selectionStyle='chartSelection3'></SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Note: To use select feature, we need to Inject **Selection** module into the **services**.

See Also

- [Display selected data for range selection](#)

Chart print in React Chart component

Print

The rendered chart can be printed directly from the browser by calling the public method print. From chart instance itself, it is called.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, LineSeries } from
'@syncfusion/ej2-react-charts';
function App() {
  const data = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];

```

```

const chartInstance;
function clickHandler() {
    chartInstance.print();
}
const primaryxAxis = { valueType: 'Category' };
return (<div>
    <button value='print' onClick={clickHandler.bind(this)}>Print</button>
    <ChartComponent id='charts' ref={chart => chartInstance = chart}
primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, Category]}/>
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='month' yName='sales'
type='Column' name='Sales'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent></div>);
}
;
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
AxesDirective, AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    const chartInstance: ChartComponent;
    function clickHandler() {
        chartInstance.print();
    }
    const primaryxAxis: AxisModel = { valueType: 'Category' };
    return (<div>
        <button value='print' onClick={clickHandler.bind(this)}>Print</button>
        <ChartComponent id='charts' ref={chart => chartInstance = chart}
primaryXAxis={primaryxAxis}>
            <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, Category]}/>
            <SeriesCollectionDirective>
                <SeriesDirective dataSource={data} xName='month' yName='sales'
type='Column' name='Sales'>
                </SeriesDirective>
            </SeriesCollectionDirective>

```



```

        </ChartComponent></div>)
    };
    export default App;
    ReactDOM.render(<App />, document.getElementById('charts'));

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
    <title>Syncfusion React Chart-Category Axis</title>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta name="description" content="Essential JS 2 for React Components"
/>
    <meta name="author" content="Syncfusion" />
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
    <script src="systemjs.config.js"></script>
    <style>
        #loader {
            color: #008cff;
            height: 40px;
            left: 45%;
            position: absolute;
            top: 45%;
            width: 30%;
        }
        #charts {
            height : 350px;
            display: block;
        }
    </style>
</head>
<body>
    <div id='charts'>
        <div id='loader'>Loading....</div>
    </div>
</body>
</html>

```

Export

The rendered chart can be exported to JPEG, PNG, SVG, PDF, XLSX, or CSV format using the export method in chart. The input parameters for this method are type for format and fileName for result.

The optional parameters for this method are,

- orientation - either portrait or landscape mode during PDF export,
- controls - pass collections of controls for multiple export,
- width - width of chart export, and
- height - height of chart export.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Export, Legend, Category, Tooltip, DataLabel, LineSeries }
from '@syncfusion/ej2-react-charts';
function App() {
    const data = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    const chartInstance;
    function clickHandler() {
        chartInstance.exportModule.export('PNG', 'sample');
    }
    const primaryXAxis = { valueType: 'Category' };
    return (<div><button value='print'
onClick={clickHandler.bind(this)}>Export</button>
    <ChartComponent id='charts' ref={chart => chartInstance = chart}
primaryXAxis={primaryXAxis}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Export,
LineSeries, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='month' yName='sales'
type='Column' name='Sales'>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent></div>);
}
;
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, AxisModel,
ColumnSeries, Export, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
}

```

```

const chartInstance: ChartComponent;
function clickHandler() {
  chartInstance.exportModule.export('PNG', 'sample');
}
const primaryxAxis: AxisModel = { valueType: 'Category' };
return (<div><button value='print'
onClick={clickHandler.bind(this)}>Export</button>
<ChartComponent id='charts' ref={chart => chartInstance = chart}
primaryXAxis={primaryxAxis}>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Export,
LineSeries, Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='month' yName='sales'
type='Column' name='Sales'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent></div>)
};
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
  <title>Syncfusion React Chart-Category Axis</title>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <meta name="description" content="Essential JS 2 for React Components"
/>
  <meta name="author" content="Syncfusion" />
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
  <script src="systemjs.config.js"></script>
  <style>
    #loader {
      color: #008cff;
      height: 40px;
      left: 45%;
      position: absolute;
      top: 45%;
      width: 30%;
    }
    #charts {
      height : 350px;
      display: block;
    }
  </style>
</head>
<body>
  <div id='charts'>
    <div id='loader'>Loading....</div>
  </div>
</body>

```

```
</html>
```

Adding header and footer in PDF export

In the export method, specify the following parameters to add a header and footer text to the exported PDF document:

- **header** - Specify the text that should appear at the top of the exported PDF document.
- **footer** - Specify the text that should appear at the bottom of the exported PDF document.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Export, Legend, Category, Tooltip, DataLabel, LineSeries }
from '@syncfusion/ej2-react-charts';
function App() {
    const data = [
        { x: 'John', y: 10000 }, { x: 'Jake', y: 12000 }, { x: 'Peter', y:
18000 },
        { x: 'James', y: 11000 }, { x: 'Mary', y: 9700 }
    ];
    let chartInstance;
    function clickHandler() {
        const header = {
            content: 'Chart Header',
            fontSize: 15
        };
        const footer = {
            content: 'Chart Footer',
            fontSize: 15,
        };
        chartInstance.exportModule.export('PDF', 'Chart', 1, [chartInstance,
null, null, true, header, footer]);
    }
    const primaryxAxis = {
        title: 'Manager',
        valueType: 'Category',
        majorGridLines: { width: 0 }
    };
    const primaryyAxis = {
        title: 'Sales',
        minimum: 0,
        maximum: 20000,
        majorGridLines: { width: 0 }
    };
    return (<div><button value='print'
onClick={clickHandler.bind(this)}>Export</button>
    <ChartComponent id='charts' ref={chart => chartInstance = chart}
primaryXAxis={primaryxAxis} primaryYAxis={primaryyAxis} title='Sales
Comparision'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Export,
LineSeries, Category]}>
```

```

        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y'
type='Column' width='2'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent></div>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
    ChartComponent, SeriesCollectionDirective, AxesDirective, AxisDirective,
    SeriesDirective, Inject, AxisModel,
    ColumnSeries, Export, Legend, Category, Tooltip, DataLabel, Zoom,
    Crosshair, LineSeries, Selection
}
    from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { x: 'John', y: 10000 }, { x: 'Jake', y: 12000 }, { x: 'Peter', y: 18000 },
        { x: 'James', y: 11000 }, { x: 'Mary', y: 9700 }
    ];
    let chartInstance: ChartComponent;
    function clickHandler() {
        const header = {
            content: 'Chart Header',
            fontSize: 15
        };
        const footer = {
            content: 'Chart Footer',
            fontSize: 15,
        };
        chartInstance.exportModule.export('PDF', 'Chart', 1, [chartInstance],
null, null, true, header, footer);
    }
    const primaryxAxis: AxisModel = {
        title: 'Manager',
        valueType: 'Category',
        majorGridLines: { width: 0 }
    };
    const primaryyAxis: AxisModel = {
        title: 'Sales',
        minimum: 0,
        maximum: 20000,
        majorGridLines: { width: 0 }
    };
    return (<div><button value='print'
onClick={clickHandler.bind(this)}>Export</button>

```

```

    <ChartComponent id='charts' ref={chart => chartInstance = chart}
    primaryXAxis={primaryxAxis} primaryYAxis={primaryyAxis} title='Sales
    Comparision'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Export,
    LineSeries, Category]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' type='Column'
    width='2'>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent></div>
    );
    export default App;
    ReactDOM.render(<App />, document.getElementById('charts'));
    {% endraw %}

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
    <title>Syncfusion React Chart-Category Axis</title>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta name="description" content="Essential JS 2 for React Components"
/>
    <meta name="author" content="Syncfusion" />
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
    <script src="systemjs.config.js"></script>
    <style>
        #loader {
            color: #008cff;
            height: 40px;
            left: 45%;
            position: absolute;
            top: 45%;
            width: 30%;
        }
        #charts {
            height : 350px;
            display: block;
        }
    </style>
</head>
<body>
    <div id='charts'>
        <div id='loader'>Loading....</div>
    </div>
</body>
</html>

```

Exporting charts into separate page during the PDF export

During PDF export, set the `exportToMultiplePage` parameter to **true** to export each chart as a separate page.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Export, Legend, Category, DateTime, Tooltip, DataLabel,
LineSeries } from '@syncfusion/ej2-react-charts';
import { AccumulationChartComponent, AccumulationSeriesCollectionDirective,
AccumulationSeriesDirective, PieSeries, AccumulationDataLabel,
AccumulationLegend } from '@syncfusion/ej2-react-charts';
function App() {
  const data1 = [
    { x: new Date(2005, 0, 1), y: 21 }, { x: new Date(2006, 0, 1), y: 24 },
    { x: new Date(2007, 0, 1), y: 36 }, { x: new Date(2008, 0, 1), y: 38 },
    { x: new Date(2009, 0, 1), y: 54 }, { x: new Date(2010, 0, 1), y: 57 },
    { x: new Date(2011, 0, 1), y: 70 }
  ];
  const data2 = [
    { x: new Date(2005, 0, 1), y: 28 }, { x: new Date(2006, 0, 1), y: 44 },
    { x: new Date(2007, 0, 1), y: 48 }, { x: new Date(2008, 0, 1), y: 50 },
    { x: new Date(2009, 0, 1), y: 66 }, { x: new Date(2010, 0, 1), y: 78 },
    { x: new Date(2011, 0, 1), y: 84 }
  ];
  const data3 = [
    { x: 'John', y: 10000 }, { x: 'Jake', y: 12000 }, { x: 'Peter', y: 18000 },
    { x: 'James', y: 11000 }, { x: 'Mary', y: 9700 }
  ];
  const data4 = [
    { x: 'Labour', y: 18, text: '18%' }, { x: 'Legal', y: 8, text: '8%' },
    { x: 'Production', y: 15, text: '15%' }, { x: 'License', y: 11, text:
'11%' },
    { x: 'Facilities', y: 18, text: '18%' }, { x: 'Taxes', y: 14, text:
'14%' },
    { x: 'Insurance', y: 16, text: '16%' }
  ];
  let chartInstance;
  let chartInstance1;
  let chartInstance2;
  function clickHandler() {
    chartInstance.exportModule.export('PDF', 'Chart', null, [chartInstance,
chartInstance1, chartInstance2], null, null, true, undefined, undefined,
true);
  }
  const primaryxAxis = {
    valueType: 'DateTime',
    labelFormat: 'y',
    intervalType: 'Years',
    edgeLabelPlacement: 'Shift',
    majorGridLines: { width: 0 }
  };
  const primaryyAxis = {
```

```

    labelFormat: '{value}%',
    rangePadding: 'None',
    minimum: 0,
    maximum: 100,
    interval: 20,
    lineStyle: { width: 0 },
    majorTickLines: { width: 0 },
    minorTickLines: { width: 0 }
  };
  const primaryxAxis1 = {
    title: 'Manager',
    valueType: 'Category',
    majorGridLines: { width: 0 }
  };
  const primaryyAxis1 = {
    title: 'Sales',
    minimum: 0,
    maximum: 20000,
    majorGridLines: { width: 0 }
  };
  const marker = { visible: true, width: 10, height: 10 };
  const datalabel = {
    visible: true,
    name: 'text',
    position: 'Inside',
    font: {
      fontWeight: '600',
      color: '#ffffff'
    }
  };
  const legendSettings = { visible: true };
  return (<div><button value='print'
onClick={clickHandler.bind(this)}>Export</button>
  <ChartComponent id='charts1' ref={chart => chartInstance = chart}
primaryXAxis={primaryxAxis} primaryYAxis={primaryyAxis} title='Medal Count'>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Export,
LineSeries, Category, DateTime]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data1} xName='x' yName='y' type='Line'
width='2' name='Germany' marker={marker}>
    </SeriesDirective>
    <SeriesDirective dataSource={data2} xName='x' yName='y' type='Line'
width='2' name='England' marker={marker}>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
  <ChartComponent id='charts2' ref={chart => chartInstance1 = chart}
primaryXAxis={primaryxAxis1} primaryYAxis={primaryyAxis1} title='Sales
Comparison'>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Export,
LineSeries, Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data3} xName='x' yName='y'
type='Column' width='2'>
    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>

```



```

    <AccumulationChartComponent id='charts' ref={chart => chartInstance2 =
    chart} title='Project Cost Breakdown' enableSmartLabels='true'
    legendSettings={legendSettings}>
      <Inject services={[PieSeries, AccumulationDataLabel,
    AccumulationLegend]} />
      <AccumulationSeriesCollectionDirective>
        <AccumulationSeriesDirective dataSource={data4} xName='x' yName='y'
    type='Pie' dataLabel={datalabel} radius='70%'>
          </AccumulationSeriesDirective>
        </AccumulationSeriesCollectionDirective>
      </AccumulationChartComponent></div>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
  ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
  AxisModel, DateTime,
  ColumnSeries, Export, Legend, Category, Tooltip, DataLabel, LineSeries,
}
  from '@syncfusion/ej2-react-charts';
import { AccumulationChartComponent, AccumulationSeriesCollectionDirective,
  AccumulationSeriesDirective, PieSeries, AccumulationDataLabel,
  AccumulationLegend } from '@syncfusion/ej2-react-charts';
function App() {
  const data1: any[] = [
    { x: new Date(2005, 0, 1), y: 21 }, { x: new Date(2006, 0, 1), y: 24 },
    { x: new Date(2007, 0, 1), y: 36 }, { x: new Date(2008, 0, 1), y: 38 },
    { x: new Date(2009, 0, 1), y: 54 }, { x: new Date(2010, 0, 1), y: 57 },
    { x: new Date(2011, 0, 1), y: 70 }
  ];
  const data2: any[] = [
    { x: new Date(2005, 0, 1), y: 28 }, { x: new Date(2006, 0, 1), y: 44 },
    { x: new Date(2007, 0, 1), y: 48 }, { x: new Date(2008, 0, 1), y: 50 },
    { x: new Date(2009, 0, 1), y: 66 }, { x: new Date(2010, 0, 1), y: 78 },
    { x: new Date(2011, 0, 1), y: 84 }
  ];
  const data3: any[] = [
    { x: 'John', y: 10000 }, { x: 'Jake', y: 12000 }, { x: 'Peter', y: 18000
  },
    { x: 'James', y: 11000 }, { x: 'Mary', y: 9700 }
  ];
  const data4: any[] = [
    { x: 'Labour', y: 18, text: '18%' }, { x: 'Legal', y: 8, text: '8%' },
    { x: 'Production', y: 15, text: '15%' }, { x: 'License', y: 11, text:
    '11%' },
    { x: 'Facilities', y: 18, text: '18%' }, { x: 'Taxes', y: 14, text:
    '14%' },
    { x: 'Insurance', y: 16, text: '16%' }
  ]
}

```

```

];
let chartInstance: ChartComponent;
let chartInstance1: ChartComponent;
let chartInstance2: ChartComponent;
function clickHandler() {
    chartInstance.exportModule.export('PDF', 'Chart', null, [chartInstance,
chartInstance1, chartInstance2], null, null, true, undefined, undefined,
true);
}
const primaryxAxis: AxisModel = {
    valueType: 'DateTime',
    labelFormat: 'Y',
    intervalType: 'Years',
    edgeLabelPlacement: 'Shift',
    majorGridLines: { width: 0 }
};
const primaryyAxis: AxisModel = {
    labelFormat: '{value}',
    rangePadding: 'None',
    minimum: 0,
    maximum: 100,
    interval: 20,
    lineStyle: { width: 0 },
    majorTickLines: { width: 0 },
    minorTickLines: { width: 0 }
};
const primaryxAxis1: AxisModel = {
    title: 'Manager',
    valueType: 'Category',
    majorGridLines: { width: 0 }
};
const primaryyAxis1: AxisModel = {
    title: 'Sales',
    minimum: 0,
    maximum: 20000,
    majorGridLines: { width: 0 }
};
const marker = { visible: true, width: 10, height: 10 };
const datalabel = {
    visible: true,
    name: 'text',
    position: 'Inside',
    font: {
        fontWeight: '600',
        color: '#ffffff'
    }
};
const legendSettings = { visible: true };
return (<div><button value='print'
onClick={clickHandler.bind(this)}>Export</button>
    <ChartComponent id='charts1' ref={chart => chartInstance = chart}
primaryXAxis={primaryxAxis} primaryYAxis={primaryyAxis} title='Medal Count'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Export,
LineSeries, Category, DateTime]} />
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={datal} xName='x' yName='y' type='Line'
width='2' name='Germany' marker={marker}>

```

```

        </SeriesDirective>
        <SeriesDirective dataSource={data2} xName='x' yName='y' type='Line'
width='2' name='England' marker={marker}>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
    <ChartComponent id='charts2' ref={chart => chartInstance1 = chart}
primaryXAxis={primaryxAxis1} primaryYAxis={primaryyAxis1} title='Sales
Comparison'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Export,
LineSeries, Category]} />
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={data3} xName='x' yName='y'
type='Column' width='2'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
    <AccumulationChartComponent id='charts' ref={chart => chartInstance2 =
chart} title='Project Cost Breakdown' enableSmartLabels='true'
legendSettings={legendSettings}>
    <Inject services={[PieSeries, AccumulationDataLabel,
AccumulationLegend]} />
    <AccumulationSeriesCollectionDirective>
        <AccumulationSeriesDirective dataSource={data4} xName='x' yName='y'
type='Pie' dataLabel={datalabel} radius='70%'>
        </AccumulationSeriesDirective>
    </AccumulationSeriesCollectionDirective>
</AccumulationChartComponent></div>
};
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
    <title>Syncfusion React Chart-Category Axis</title>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta name="description" content="Essential JS 2 for React Components"
/>
    <meta name="author" content="Syncfusion" />
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
    <script src="systemjs.config.js"></script>
    <style>
        #loader {
            color: #008cff;
            height: 40px;
            left: 45%;
            position: absolute;
            top: 45%;
            width: 30%;

```

```

    }
    #charts {
        height : 350px;
        display: block;
    }
</style>
</head>
<body>
    <div id='charts'>
        <div id='loader'>Loading....</div>
    </div>
</body>
</html>

```

Multiple chart export

You can export the multiple charts in single page by passing the multiple chart objects in the export method of chart. To export multiple charts in a single page, follow the given steps:

Initially, render more than one chart to export, and then add button to export the multiple charts. In button click, call the export method in charts, and then pass the multiple chart objects in the export method.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Export, Legend, Category, Tooltip, DataLabel, LineSeries }
from '@syncfusion/ej2-react-charts';
function App() {
    const data = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    const data1 = [
        { month: 'jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    const chartInstance;
    const chartInstance1;
    function clickHandler() {
        chartInstance.exportModule.export('PNG', 'sample', null,
[chartInstance, chartInstance1]);
    }
    const primaryxAxis = { valueType: 'Category' };
    return (<div><button value='print'
onClick={clickHandler.bind(this)}>Export</button>

```

```

    <ChartComponent id='charts1' ref={chart => chartInstance = chart}
    primaryXAxis={primaryxAxis}>
      <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Export,
    LineSeries, Category]}/>
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='month' yName='sales'
    type='Column' name='Sales'>
          </SeriesDirective>
        </SeriesCollectionDirective>
      </ChartComponent>
      <ChartComponent id='charts2' ref={chart => chartInstance1 = chart}
    primaryXAxis={primaryxAxis}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel, Export,
    LineSeries, Category]}/>
        <SeriesCollectionDirective>
          <SeriesDirective dataSource={data1} xName='month' yName='sales'
    type='Column' name='Sales'>
            </SeriesDirective>
          </SeriesCollectionDirective>
        </ChartComponent> </div>;
    }
    ;
    export default App;
    ReactDOM.render(<App />, document.getElementById('charts'));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, AxisModel,
ColumnSeries, Export, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection}
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const data1: any[] = [
    { month: 'jan', sales: 35 }, { month: 'Feb', sales: 28 },
    { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
    { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
    { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
    { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
    { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
  ];
  const chartInstance: ChartComponent;
  const chartInstance1: ChartComponent;
  function clickHandler() { chartInstance.exportModule.export('PNG',
'sample', null, [chartInstance, chartInstance1]);
  }
}

```

```

const primaryxAxis: AxisModel = { valueType: 'Category' };
return (<div><button value='print'
onClick={clickHandler.bind(this)}>Export</button>
    <ChartComponent id='charts1' ref={chart => chartInstance = chart}
primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,Export,
LineSeries, Category]} />
    <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='month' yName='sales'
type='Column' name='Sales'>
    </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
    <ChartComponent id='charts2' ref={chart => chartInstance1 = chart}
primaryXAxis={primaryxAxis}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,Export,
LineSeries, Category]} />
    <SeriesCollectionDirective>
    <SeriesDirective dataSource={data1} xName='month' yName='sales'
type='Column' name='Sales'>
    </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent> </div>)
);
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
    <title>Syncfusion React Chart-Category Axis</title>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta name="description" content="Essential JS 2 for React Components"
/>
    <meta name="author" content="Syncfusion" />
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
    <script src="systemjs.config.js"></script>
    <style>
        #loader {
            color: #008cff;
            height: 40px;
            left: 45%;
            position: absolute;
            top: 45%;
            width: 30%;
        }
        #charts {
            height : 350px;
            display: block;
        }
    </style>

```

```

</head>
<body>
  <div id='charts'>
    <div id='loader'>Loading....</div>
  </div>
</body>
</html>

```

Exporting chart using base64 string

The chart can be exported as an image in the form of a base64 string by utilizing HTML canvas. This process involves rendering the chart onto a canvas element and then converting the canvas content to a base64 string.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Category } from '@syncfusion/ej2-react-charts';
function App() {
  const data = [
    { x: 'DEU', y: 35.5 }, { x: 'CHN', y: 18.3 }, { x: 'ITA', y: 17.6 }, {
x: 'JPN', y: 13.6 },
    { x: 'US', y: 12 }, { x: 'ESP', y: 5.6 }, { x: 'FRA', y: 4.6 }, { x:
'AUS', y: 3.3 },
    { x: 'BEL', y: 3 }, { x: 'UK', y: 2.9 }
  ];
  let chartInstance;
  function clickHandler() {
    var svg = document.querySelector("#charts_svg");
    var svgData = new XMLSerializer().serializeToString(svg);
    var canvas = document.createElement("canvas");
    document.body.appendChild(canvas);
    var svgSize = svg.getBoundingClientRect();
    canvas.width = svgSize.width;
    canvas.height = svgSize.height;
    var ctx = canvas.getContext("2d");
    var img = document.createElement("img");
    img.setAttribute("src", "data:image/svg+xml;base64," + btoa(svgData));
    img.onload = function() {
      ctx.drawImage(img, 0, 0);
      var imagedata = canvas.toDataURL("image/png");
      console.log(imagedata); // printed base64 in console
      canvas.remove();
    };
  }
  return (<div><ChartComponent id="charts" style={{ textAlign: 'center' }}
primaryXAxis={{
  valueType: 'Category',
  majorGridLines: { width: 0 },
  majorTickLines: { width: 0 },
  minorTickLines: { width: 0 }
}} primaryYAxis={{
  minimum: 0,
  maximum: 40,

```

```

        interval: 10,
        lineStyle: {width : 0},
        minorTickLines: {width: 0},
        majorTickLines: {width : 0},
    }} chartArea={{ border: { width: 0 } }}>
    <Inject services={[ColumnSeries, Category]}/>
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName="x" yName="y"
type="Column"></SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>

<button value='Export'
onClick={clickHandler.bind(this)}>Export</button></div>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Category } from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { x: 'DEU', y: 35.5 }, { x: 'CHN', y: 18.3 }, { x: 'ITA', y: 17.6 }, {
x: 'JPN', y: 13.6 },
        { x: 'US', y: 12 }, { x: 'ESP', y: 5.6 }, { x: 'FRA', y: 4.6 }, { x:
'AUS', y: 3.3 },
        { x: 'BEL', y: 3 }, { x: 'UK', y: 2.9 }
    ];
    let chartInstance: ChartComponent;
    function clickHandler() {
        var svg = document.querySelector("#charts_svg");
        var svgData = new XMLSerializer().serializeToString(svg);
        var canvas = document.createElement("canvas");
        document.body.appendChild(canvas);
        var svgSize = svg.getBoundingClientRect();
        canvas.width = svgSize.width;
        canvas.height = svgSize.height;
        var ctx = canvas.getContext("2d");
        var img = document.createElement("img");
        img.setAttribute("src", "data:image/svg+xml;base64," +
btoa(svgData));
        img.onload = function() {
            ctx.drawImage(img, 0, 0);
            var imagedata = canvas.toDataURL("image/png");
            console.log(imagedata); // printed base64 in console
            canvas.remove();
        };
    }
}

```



```

return (<div> <ChartComponent id="charts" style={{ textAlign: 'center' }}
primaryXAxis={{
  valueType: 'Category',
  majorGridLines: { width: 0 },
  majorTickLines: { width: 0 },
  minorTickLines: { width: 0 }
}} primaryYAxis={{
  minimum: 0,
  maximum: 40,
  interval: 10,
  lineStyle: {width : 0},
  minorTickLines: {width: 0},
  majorTickLines: {width : 0},
}} chartArea={{ border: { width: 0 } }}>
  <Inject services={[ColumnSeries, Category]}>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName="x" yName="y"
type="Column"></SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
  <button value='Export'
onClick={clickHandler.bind(this)}>Export</button></div>
);
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% enddraw %}

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
  <title>Syncfusion React Chart-Category Axis</title>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <meta name="description" content="Essential JS 2 for React Components"
/>
  <meta name="author" content="Syncfusion" />
  <script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
  <script src="systemjs.config.js"></script>
  <style>
    #loader {
      color: #008cff;
      height: 40px;
      left: 45%;
      position: absolute;
      top: 45%;
      width: 30%;
    }
    #charts {
      height : 350px;
      display: block;
    }
  </style>

```

```

</head>
<body>
  <div id='charts'>
    <div id='loader'>Loading....</div>
  </div>
</body>
</html>

```

Chart appearance in React Chart component

Custom color palette

You can customize the default color of series or points by providing a custom color palette of your choice by using the [palettes](#) property.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
function App() {
  const data = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  const palette = ["#E94649", "#F6B53F", "#6FAAB0", "#C4C24A"];
  const primaryxAxis = { valueType: 'Category', title: 'Countries' };
  const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} palettes={palette} title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' legendShape='Circle' type='Column'>
    </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='silver'
type='Column' name='Silver' legendShape='Rectangle'>
    </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze' legendShape='Rectangle' type='Column'>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, Legend, Category, Tooltip, DataLabel, ColumnSeries}
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { country: "USA", gold: 50, silver: 70, bronze: 45 },
    { country: "China", gold: 40, silver: 60, bronze: 55 },
    { country: "Japan", gold: 70, silver: 60, bronze: 50 },
    { country: "Australia", gold: 60, silver: 56, bronze: 40 },
    { country: "France", gold: 50, silver: 45, bronze: 35 },
    { country: "Germany", gold: 40, silver: 30, bronze: 22 },
    { country: "Italy", gold: 40, silver: 35, bronze: 37 },
    { country: "Sweden", gold: 30, silver: 25, bronze: 27 }
  ];
  const palette: string[] = ["#E94649", "#F6B53F", "#6FAAB0", "#C4C24A"];
  const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
  const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    palettes={palette}
    title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold'
        legendShape='Circle' type='Column'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='silver'
type='Column'
        name='Silver' legendShape='Rectangle'>
      </SeriesDirective>
      <SeriesDirective dataSource={data} xName='country' yName='bronze'
name='Bronze'
        legendShape='Rectangle' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Data point customization

The color of individual data point or data points within a range can be customized using the options below.

Point color mapping

You can bind the color for the points from [dataSource](#) for the series using [pointColorMapping](#) property.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Category } from '@syncfusion/ej2-react-charts';
function App() {
  const data = [
    { x: 'Jan', y: 6.96, color: "red" },
    { x: 'Feb', y: 8.9, color: "blue" },
    { x: 'Mar', y: 12, color: "orange" },
    { x: 'Apr', y: 17.5, color: "aqua" },
    { x: 'May', y: 22.1, color: "grey" }
  ];
  return <ChartComponent id='charts' primaryXAxis={{ valueType:
'Category', majorGridLines: { width: 0 } }} primaryYAxis={{ lineStyle: {
width: 0 },
majorTickLines: { width: 0 },
minorTickLines: { width: 0 },
labelFormat: '{value}°C' }} title="USA CLIMATE - WEATHER BY
MONTH" chartArea={{ border: { width: 0 } }}>
    <Inject services={[ColumnSeries, Category]}>
    <SeriesCollectionDirective>
      <SeriesDirective pointColorMapping="color"
dataSource={data} name='USA' xName='x' yName='y' type='Column' animation={{
enable: false }} cornerRadius={{
topLeft: 10, topRight: 10
}}>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Category, Selection, Legend, RangeColorSettingsDirective,
RangeColorSettingDirective } from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { x: 'Jan', y: 6.96, color: "red" },
    { x: 'Feb', y: 8.9, color: "blue" },
    { x: 'Mar', y: 12, color: "orange" },
    { x: 'Apr', y: 17.5, color: "aqua" },
    { x: 'May', y: 22.1, color: "grey" }
  ]
}
```

```

];
return <ChartComponent id='charts' primaryXAxis={{ valueType: 'Category',
majorGridLines: { width: 0 } }} primaryYAxis={{ lineStyle: { width: 0 },
majorTickLines: { width: 0 },
minorTickLines: { width: 0 },
labelFormat: '{value}°C' }} title="USA CLIMATE - WEATHER BY
MONTH" chartArea={{ border: { width: 0 } }}>
    <Inject services={[ColumnSeries, Category]}>
    <SeriesCollectionDirective>
        <SeriesDirective pointColorMapping="color"
dataSource={data} name='USA' xName='x' yName='y' type='Column' animation={{
enable: false }} cornerRadius={{
    topLeft: 10, topRight: 10
}}>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Range color mapping

You can differentiate data points based on their y values using [rangeColorSettings](#) in the chart.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Category, Selection, Legend, RangeColorSettingsDirective,
RangeColorSettingDirective } from '@syncfusion/ej2-react-charts';
function App() {
    const data = [
        { x: "Jan", y: 6.96 },
        { x: "Feb", y: 8.9 },
        { x: "Mar", y: 12 },
        { x: "Apr", y: 17.5 },
        { x: "May", y: 22.1 },
        { x: "June", y: 25 },
        { x: "July", y: 29.4 },
        { x: "Aug", y: 29.6 },
        { x: "Sep", y: 25.8 },
        { x: "Oct", y: 21.1 },
        { x: "Nov", y: 15.5 },
        { x: "Dec", y: 9.9 }
    ];
    const color1 = ['#F9D422'];
    const color2 = ['#F28F3F'];
    const color3 = ['#E94F53'];
    return <ChartComponent id='charts' style={{ textAlign: "center" }}
selectionMode='Point' primaryXAxis={{ valueType: 'Category', majorGridLines:
{ width: 0 } }} primaryYAxis={{ lineStyle: { width: 0 },
majorTickLines: { width: 0 },
minorTickLines: { width: 0 },

```

```

        labelFormat: '{value}°C' }} title="USA CLIMATE - WEATHER BY
MONTH" chartArea={{ border: { width: 0 } }} legendSettings={{
    mode: 'Range',
    visible: true,
    toggleVisibility: false
  }}>
      <Inject services={[ColumnSeries, Selection,
Category, Legend]}/>
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} name='USA'
xName='x' yName='y' type='Column' animation={{ enable: false }}
cornerRadius={{
    topLeft: 10, topRight: 10
  }}>
      </SeriesDirective>
    </SeriesCollectionDirective>
    <RangeColorSettingsDirective>
      <RangeColorSettingDirective label="1°C to 10°C"
start={1} end={10} colors={color1}></RangeColorSettingDirective>
      <RangeColorSettingDirective label="11°C to 20°C"
start={11} end={20} colors={color2}></RangeColorSettingDirective>
      <RangeColorSettingDirective label="21°C to 30°C"
start={21} end={30} colors={color3}></RangeColorSettingDirective>
    </RangeColorSettingsDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Category, Selection, Legend, RangeColorSettingsDirective,
RangeColorSettingDirective } from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { x: "Jan", y: 6.96 },
    { x: "Feb", y: 8.9 },
    { x: "Mar", y: 12 },
    { x: "Apr", y: 17.5 },
    { x: "May", y: 22.1 },
    { x: "June", y: 25 },
    { x: "July", y: 29.4 },
    { x: "Aug", y: 29.6 },
    { x: "Sep", y: 25.8 },
    { x: "Oct", y: 21.1 },
    { x: "Nov", y: 15.5 },
    { x: "Dec", y: 9.9 }
  ];
  const color1: string[] = ['#F9D422'];
  const color2: string[] = ['#F28F3F'];

```

```

const color3: string[] = ['#E94F53'];

return <ChartComponent id='charts' style={{ textAlign: "center" }}
selectionMode='Point' primaryXAxis={{ valueType: 'Category', majorGridLines:
{ width: 0 } }} primaryYAxis={{ lineStyle: { width: 0 },
majorTickLines: { width: 0 },
minorTickLines: { width: 0 },
labelFormat: '{value}°C' }} title="USA CLIMATE - WEATHER BY
MONTH" chartArea={{ border: { width: 0 } }} legendSettings={{
mode: 'Range',
visible: true,
toggleVisibility: false
}}>
    <Inject services={[ColumnSeries, Selection,
Category, Legend]}/>
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} name='USA'
xName='x' yName='y' type='Column' animation={{ enable: false }}
cornerRadius={{
    topLeft: 10, topRight: 10
}}>
            </SeriesDirective>
        </SeriesCollectionDirective>
        <RangeColorSettingsDirective>
            <RangeColorSettingDirective label="1°C to 10°C"
start={1} end={10} colors={color1}></RangeColorSettingDirective>
            <RangeColorSettingDirective label="11°C to 20°C"
start={11} end={20} colors={color2}></RangeColorSettingDirective>
            <RangeColorSettingDirective label="21°C to 30°C"
start={21} end={30} colors={color3}></RangeColorSettingDirective>
        </RangeColorSettingsDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% enddraw %}

```

Point level customization

Marker, datalabel and fill color of each data point can be customized with [pointRender](#) and [textRender](#) event.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
function App() {
    const data = [
        { country: "USA", gold: 50 },
        { country: "China", gold: 40 },
        { country: "Japan", gold: 70 },
        { country: "Australia", gold: 60 },
        { country: "France", gold: 50 },
        { country: "Germany", gold: 40 },
    ];

```

```

        { country: "Italy", gold: 40 },
        { country: "Sweden", gold: 30 }
    ];
    const pointRender = (args) => {
        let seriesColor = ['#00bdae', '#404041', '#357cd2', '#e56590',
            '#f8b883',
            '#70ad47', '#dd8abd', '#7f84e8', '#7bb4eb', '#ea7a57'];
        args.fill = seriesColor[args.point.index];
    };
    const primaryxAxis = { valueType: 'Category', title: 'Countries' };
    const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} pointRender={pointRender} title='Olympic
Medals'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' type='Column'>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
    }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, Legend, Category, Tooltip, DataLabel,
ColumnSeries, IPointRenderEventArgs } from '@syncfusion/ej2-react-charts';
import { EmitType } from '@syncfusion/ej2-charts';
function App() {
    const data: any[] = [
        { country: "USA", gold: 50 },
        { country: "China", gold: 40 },
        { country: "Japan", gold: 70 },
        { country: "Australia", gold: 60 },
        { country: "France", gold: 50 },
        { country: "Germany", gold: 40 },
        { country: "Italy", gold: 40 },
        { country: "Sweden", gold: 30 }
    ];
    const pointRender: EmitType<IPointRenderEventArgs> = (args:
IPointRenderEventArgs): void => {
        let seriesColor: string[] = ['#00bdae', '#404041', '#357cd2', '#e56590',
            '#f8b883',
            '#70ad47', '#dd8abd', '#7f84e8', '#7bb4eb', '#ea7a57'];
        args.fill = seriesColor[args.point.index];
    };
    const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };

```



```

const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
return <ChartComponent id='charts'
  primaryXAxis={primaryxAxis}
  primaryYAxis={primaryyAxis}
  pointRender={pointRender}
  title='Olympic Medals'>
  <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
</>;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

<!-- markdownlint-disable MD036 -->

Chart area customization

<!-- markdownlint-disable MD036 -->

Customize the chart background

<!-- markdownlint-disable MD013 -->

Using [background](#) and [border](#) properties, you can change the background color and border of the chart.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
function App() {
  const data = [
    { country: "USA", gold: 50 },
    { country: "China", gold: 40 },
    { country: "Japan", gold: 70 },
    { country: "Australia", gold: 60 },
    { country: "France", gold: 50 },
    { country: "Germany", gold: 40 },
    { country: "Italy", gold: 40 },
    { country: "Sweden", gold: 30 }
  ];
  const primaryxAxis = { valueType: 'Category', title: 'Countries' };
  const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
  const border = { width: 2, color: '#FF0000' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} background='skyblue' border={border}
title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
  </>
}

```

```

        <SeriesCollectionDirective>
          <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' type='Column'>
            </SeriesDirective>
          </SeriesCollectionDirective>
        </ChartComponent>;
      }
    }
    ;
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, BorderModel, Legend, Category, Tooltip, DataLabel,
ColumnSeries } from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { country: "USA", gold: 50 },
    { country: "China", gold: 40 },
    { country: "Japan", gold: 70 },
    { country: "Australia", gold: 60 },
    { country: "France", gold: 50 },
    { country: "Germany", gold: 40 },
    { country: "Italy", gold: 40 },
    { country: "Sweden", gold: 30 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
  const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
  const border: BorderModel = { width: 2, color: '#FF0000' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    background='skyblue' border={border}
    title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  </>;
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));

```

Chart margin

You can set margin for chart from its container through [margin](#) property.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
function App() {
    const data = [
        { country: "USA", gold: 50 },
        { country: "China", gold: 40 },
        { country: "Japan", gold: 70 },
        { country: "Australia", gold: 60 },
        { country: "France", gold: 50 },
        { country: "Germany", gold: 40 },
        { country: "Italy", gold: 40 },
        { country: "Sweden", gold: 30 }
    ];
    const primaryxAxis = { valueType: 'Category', title: 'Countries' };
    const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
    const border = { width: 2, color: '#FF0000' };
    const margin = { left: 40, right: 40, top: 40, bottom: 40 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} background='skyblue' border={border}
margin={margin} title='Olympic Medals'>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' type='Column'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, BorderModel, MarginModel, Legend, Category, Tooltip,
DataLabel, ColumnSeries } from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { country: "USA", gold: 50 },
        { country: "China", gold: 40 },
        { country: "Japan", gold: 70 },
        { country: "Australia", gold: 60 },
        { country: "France", gold: 50 },
        { country: "Germany", gold: 40 },
        { country: "Italy", gold: 40 },
        { country: "Sweden", gold: 30 }
    ];
};

```

```

const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
const border: BorderModel = { width: 2, color: '#FF0000' };
const margin: MarginModel = { left: 40, right: 40, top: 40, bottom: 40 };
return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    background='skyblue' border={border}
    margin={margin}
    title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' type='Column'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Chart area customization

Using [background](#) and [border](#) properties, you can change the background color and border of the chart area. Width for the chart area can be customized using [width](#) property.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
function App() {
    const data = [
        { country: "USA", gold: 50 },
        { country: "China", gold: 40 },
        { country: "Japan", gold: 70 },
        { country: "Australia", gold: 60 },
        { country: "France", gold: 50 },
        { country: "Germany", gold: 40 },
        { country: "Italy", gold: 40 },
        { country: "Sweden", gold: 30 }
    ];
    const primaryxAxis = { valueType: 'Category', title: 'Countries' };
    const primaryyAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
    const border = { width: 2, color: '#FF0000' };
    const chartarea = { background: 'skyblue', width: '90%' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} border={border} chartArea={chartarea}
title='Olympic Medals'>

```

```

    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, BorderModel, ChartAreaModel, Legend, Category,
Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { country: "USA", gold: 50 },
    { country: "China", gold: 40 },
    { country: "Japan", gold: 70 },
    { country: "Australia", gold: 60 },
    { country: "France", gold: 50 },
    { country: "Germany", gold: 40 },
    { country: "Italy", gold: 40 },
    { country: "Sweden", gold: 30 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
  const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
title: 'Medals' };
  const border: BorderModel = { width: 2, color: '#FF0000' };
  const chartarea: ChartAreaModel = { background: 'skyblue', width: '90%' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    border={border}
    chartArea={chartarea}
    title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' type='Column'>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  </>;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Animation

You can customize animation for a particular series using [animation](#) property. You can enable or disable animation of the series using [enable](#) property, [duration](#) specifies the duration of an animation and [delay](#) allows us to start the animation at desire time.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
react-charts';
function App() {
  const data = [
    { country: "USA", gold: 50 },
    { country: "China", gold: 40 },
    { country: "Japan", gold: 70 },
    { country: "Australia", gold: 60 },
    { country: "France", gold: 50 },
    { country: "Germany", gold: 40 },
    { country: "Italy", gold: 40 },
    { country: "Sweden", gold: 30 }
  ];
  const primaryXAxis = { valueType: 'Category', title: 'Countries' };
  const primaryYAxis = { minimum: 0, maximum: 80, interval: 20, title:
'Medals' };
  const border = { width: 2, color: 'grey' };
  const animation = { enable: true, duration: 1200, delay: 100 };
  return <ChartComponent id='charts' primaryXAxis={primaryXAxis}
primaryYAxis={primaryYAxis} title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
name='Gold' border={border} animation={animation} type='Column'>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, Legend, Category, Tooltip, DataLabel, ColumnSeries}
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { country: "USA", gold: 50 },
    { country: "China", gold: 40 },
    { country: "Japan", gold: 70 },
    { country: "Australia", gold: 60 },
  ];
```

```

    { country: "France", gold: 50 },
    { country: "Germany", gold: 40 },
    { country: "Italy", gold: 40 },
    { country: "Sweden", gold: 30 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category', title:
  'Countries' };
  const primaryyAxis: AxisModel = { minimum: 0, maximum: 80, interval: 20,
  title: 'Medals' };
  const border = { width: 2, color: 'grey' };
  const animation = { enable: true, duration: 1200, delay: 100 };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Olympic Medals'>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
  Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='country' yName='gold'
  name='Gold'
        border={border}
        animation={animation} type='Column'>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));

```

Fluid animation

Fluid animation used to animate series with updated dataSource continues animation rather than animation whole series. You can customize animation for a particular series using `animate` method.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
  Legend, Category, Tooltip, DataLabel, ColumnSeries } from '@syncfusion/ej2-
  react-charts';
function App() {
  const data = [
    { x: 'Egg', y: 106 },
    { x: 'Fish', y: 103 },
    { x: 'Misc', y: 198 },
    { x: 'Tea', y: 189 },
    { x: 'Fruits', y: 250 }
  ];
  const primaryxAxis = { valueType: 'Category', interval: 1, tickPosition:
  'Inside',
    labelPosition: 'Inside', labelStyle: { color: '#ffffff' } };
  const primaryyAxis = { minimum: 0, maximum: 300, interval: 50,
  labelStyle: { color: 'transparent' } };
  let count = 0;
  function onChartLoad(args) {

```

```

let chart = document.getElementById('charts');
chart.setAttribute('title', '');
args.chart.loaded = null;
let columninterval = setInterval(() => {
    if (document.getElementById('charts')) {
        if (count === 0) {
            args.chart.series[0].dataSource = [
                { x: 'Egg', y: 206 },
                { x: 'Fish', y: 123 },
                { x: 'Misc', y: 48 },
                { x: 'Tea', y: 240 },
                { x: 'Fruits', y: 170 }
            ];
            args.chart.animate();
            count++;
        }
        else if (count === 1) {
            args.chart.series[0].dataSource = [
                { x: 'Egg', y: 86 },
                { x: 'Fish', y: 173 },
                { x: 'Misc', y: 188 },
                { x: 'Tea', y: 109 },
                { x: 'Fruits', y: 100 }
            ];
            args.chart.animate();
            count++;
        }
        else if (count === 2) {
            args.chart.series[0].dataSource = [
                { x: 'Egg', y: 156 },
                { x: 'Fish', y: 33 },
                { x: 'Misc', y: 260 },
                { x: 'Tea', y: 200 },
                { x: 'Fruits', y: 30 }
            ];
            args.chart.animate();
            count = 0;
        }
    }
    else {
        clearInterval(columninterval);
    }
}, 2000);

return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Olympic Medals'
loaded={onChartLoad.bind(this)}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]}/>
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} type='Column'
xName='x' width={2} yName='y' name='Tiger' cornerRadius={{ bottomLeft: 10,
bottomRight: 10, topLeft: 10, topRight: 10 }} marker={{ dataLabel: {
visible: true, position: 'Top', font: { fontWeight: '600', color: '#ffffff'
} } }}>
        </SeriesDirective>
    </SeriesCollectionDirective>

```



```

    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, Legend, Category, Tooltip, DataLabel, ColumnSeries}
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { x: 'Egg', y: 106 },
    { x: 'Fish', y: 103 },
    { x: 'Misc', y: 198 },
    { x: 'Tea', y: 189 },
    { x: 'Fruits', y: 250 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category', interval: 1,
tickPosition: 'Inside',
    labelPosition: 'Inside', labelStyle: { color:
'#ffffff' } };
  const primaryyAxis: AxisModel = { minimum: 0, maximum: 300, interval: 50,
labelStyle: { color: 'transparent' } };
  let count = 0;
  function onChartLoad(args: ILoadedEventArgs): void {
    let chart: Element = document.getElementById('charts');
    chart.setAttribute('title', '');
    args.chart.loaded = null;
    let columninterval = setInterval(
      () => {
        if (document.getElementById('charts')) {
          if (count === 0) {
            args.chart.series[0].dataSource = [
              { x: 'Egg', y: 206 },
              { x: 'Fish', y: 123 },
              { x: 'Misc', y: 48 },
              { x: 'Tea', y: 240 },
              { x: 'Fruits', y: 170 }
            ];
            args.chart.animate();
            count++;
          }
          else if (count === 1) {
            args.chart.series[0].dataSource = [
              { x: 'Egg', y: 86 },
              { x: 'Fish', y: 173 },
              { x: 'Misc', y: 188 },
              { x: 'Tea', y: 109 },
              { x: 'Fruits', y: 100 }
            ];

```

```

        args.chart.animate();
        count++;
    }
    else if (count === 2) {
        args.chart.series[0].dataSource = [
            { x: 'Egg', y: 156 },
            { x: 'Fish', y: 33 },
            { x: 'Misc', y: 260 },
            { x: 'Tea', y: 200 },
            { x: 'Fruits', y: 30 }
        ];
        args.chart.animate();
        count = 0;
    }
    } else {
        clearInterval(columninterval);
    }
    },
    2000
);
}

return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Olympic Medals' loaded={onChartLoad.bind(this)}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} type='Column'
xName='x' width={2} yName='y' name='Tiger'
            cornerRadius={{ bottomLeft: 10, bottomRight:
10, topLeft: 10, topRight: 10 }}
            marker={{ dataLabel: { visible: true,
position: 'Top', font: { fontWeight: '600', color: '#ffffff' } } }}>
        </SeriesDirective>
    </SeriesCollectionDirective>

    </ChartComponent>

};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% enddraw %}

```

Chart title

Chart can be given a title using [title](#) property, to show the information about the data plotted.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, StepLineSeries } from
'@syncfusion/ej2-react-charts';
function App() {
    const data = [

```

```

        { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
        { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
        { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
        { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
        { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
        { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
        { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
        { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
    ];
    const primaryxAxis = { title: 'Years', lineStyle: { width: 0 },
labelFormat: 'Y',
        intervalType: 'Years', valueType: 'DateTime', edgeLabelPlacement:
'Shift' };
    const primaryyAxis = { title: 'Percentage (%)', labelFormat: '{value}%',
        minimum: 0, maximum: 20, interval: 2 };
    const titlestyle = { fontFamily: "Arial", fontStyle: 'italic',
fontWeight: 'regular',
        color: "#E27F2D", size: '23px' };
    const marker = { visible: true, width: 10, height: 10 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Unemployment Rates 1975-2010'
titleStyle={titlestyle}>
        <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
DateTime]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y'
name='China' width={2} type='StepLine' marker={marker}></SeriesDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y1'
name='Australia' width={2} type='StepLine'
marker={marker}></SeriesDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y2'
name='Japan' width={2} type='StepLine' marker={marker}></SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, FontModel, Legend, DateTime, Tooltip, DataLabel,
StepLineSeries } from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
        { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
        { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
        { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
        { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
        { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
        { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
        { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
    ];

```

```

    ];
    const primaryxAxis: AxisModel= {title: 'Years',lineStyle: { width: 0
},labelFormat: 'Y',
    intervalType: 'Years',valueType: 'DateTime',edgeLabelPlacement:
'Shift'} ;
    const primaryyAxis: AxisModel= {title: 'Percentage (%)',labelFormat:
'{{value}}%',
    minimum: 0, maximum: 20, interval: 2} ;
    const titlestyle:FontModel ={fontFamily: "Arial", fontStyle:
'italic',fontWeight: 'regular',
    color: "#E27F2D", size: '23px'};
    const marker={ visible: true, width: 10, height: 10 };
    return <ChartComponent id='charts'
        primaryXAxis={ primaryxAxis }
        primaryYAxis={ primaryyAxis }
        title='Unemployment Rates 1975-2010'
        titleStyle = { titlestyle }>
        <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
DateTime]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource ={data} xName='x' yName='y'
name='China' width={2}type='StepLine' marker={ marker }></SeriesDirective>
            <SeriesDirective dataSource ={data} xName='x' yName='y1'
name='Australia' width={2}
                type='StepLine' marker={ marker }></SeriesDirective>
            <SeriesDirective dataSource ={data} xName='x' yName='y2'
name='Japan' width={2}type='StepLine' marker={ marker }></SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Title position

By using the [position](#) property in [titleStyle](#), you can position the [title](#) at left, right, top or bottom of the chart. The title is positioned at the top of the chart, by default.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, StepLineSeries } from
'@syncfusion/ej2-react-charts';
function App() {
    const data = [
        { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
        { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
        { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
        { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
        { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
        { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
        { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
        { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
    ];

```

```

const primaryxAxis = {
  title: 'Years', lineStyle: { width: 0 }, labelFormat: 'y',
  intervalType: 'Years', valueType: 'DateTime', edgeLabelPlacement:
'Shift'
};
const primaryyAxis = {
  title: 'Percentage (%)', labelFormat: '{value}%',
  minimum: 0, maximum: 20, interval: 2
};
const titlestyle = { position: 'Bottom' };
const marker = { visible: true, width: 10, height: 10 };
return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Unemployment Rates 1975-2010'
titleStyle={titlestyle} legendSettings={{ visible: false }}>
  <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
DateTime]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y'
name='China' width={2} type='StepLine' marker={marker}></SeriesDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y1'
name='Australia' width={2} type='StepLine'
marker={marker}></SeriesDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y2'
name='Japan' width={2} type='StepLine' marker={marker}></SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, FontModel, Legend, DateTime, Tooltip, DataLabel,
StepLineSeries } from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
    { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
    { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
    { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
    { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
    { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
    { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
    { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
  ];
  const primaryxAxis: AxisModel = {
    title: 'Years', lineStyle: { width: 0 }, labelFormat: 'y',
    intervalType: 'Years', valueType: 'DateTime',
edgeLabelPlacement: 'Shift'
  };

```

```

const primaryyAxis: AxisModel = {
    title: 'Percentage (%)', labelFormat: '{value}%',
    minimum: 0, maximum: 20, interval: 2
};
const titlestyle: FontModel = { position: 'Bottom' };
const marker = { visible: true, width: 10, height: 10 };
return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Unemployment Rates 1975-2010'
    titleStyle={titlestyle}
    legendSettings={{ visible: false }}>
    <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
DateTime]} />
    <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y'
name='China' width={2} type='StepLine' marker={marker}></SeriesDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y1'
name='Australia' width={2}
            type='StepLine'
marker={marker}></SeriesDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y2'
name='Japan' width={2} type='StepLine' marker={marker}></SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

The custom option helps you to position the title anywhere in the chart using [x](#) and [y](#) coordinates.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, StepLineSeries } from
'@syncfusion/ej2-react-charts';
function App() {
    const data = [
        { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
        { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
        { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
        { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
        { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
        { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
        { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
        { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
    ];
    const primaryxAxis = {
        title: 'Years', lineStyle: { width: 0 }, labelFormat: 'y',
        intervalType: 'Years', valueType: 'DateTime', edgeLabelPlacement:
'Shift'
    };

```

```

const primaryyAxis = {
  title: 'Percentage (%)', labelFormat: '{value}%',
  minimum: 0, maximum: 20, interval: 2
};
const titlestyle = { position: 'Custom', x: 300, y: 60 };
const marker = { visible: true, width: 10, height: 10 };
return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Unemployment Rates 1975-2010'
titleStyle={titlestyle}>
  <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
DateTime]} />
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y'
name='China' width={2} type='StepLine' marker={marker}></SeriesDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y1'
name='Australia' width={2} type='StepLine'
marker={marker}></SeriesDirective>
    <SeriesDirective dataSource={data} xName='x' yName='y2'
name='Japan' width={2} type='StepLine' marker={marker}></SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% enddraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, FontModel, Legend, DateTime, Tooltip, DataLabel,
StepLineSeries } from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
    { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
    { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
    { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
    { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
    { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
    { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
    { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
  ];
  const primaryxAxis: AxisModel = {
    title: 'Years', lineStyle: { width: 0 }, labelFormat: 'y',
    intervalType: 'Years', valueType: 'DateTime',
edgeLabelPlacement: 'Shift'
  };
  const primaryyAxis: AxisModel = {
    title: 'Percentage (%)', labelFormat: '{value}%',
    minimum: 0, maximum: 20, interval: 2
  };
  const titlestyle: FontModel = { position: 'Custom', x: 300, y: 60 };

```

```

    const marker = { visible: true, width: 10, height: 10 };
    return <ChartComponent id='charts'
      primaryXAxis={primaryxAxis}
      primaryYAxis={primaryyAxis}
      title='Unemployment Rates 1975-2010'
      titleStyle={titlestyle}>
      <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
DateTime]} />
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y'
name='China' width={2} type='StepLine' marker={marker}></SeriesDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y1'
name='Australia' width={2}
          type='StepLine'
marker={marker}></SeriesDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y2'
name='Japan' width={2} type='StepLine' marker={marker}></SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));
  {% endraw %}

```

Title alignment

You can align the title to the near, far, or center of the chart using the [textAlignment](#) property.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, StepLineSeries } from
'@syncfusion/ej2-react-charts';
function App() {
  const data = [
    { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
    { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
    { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
    { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
    { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
    { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
    { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
    { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
  ];
  const primaryxAxis = {
    title: 'Years', lineStyle: { width: 0 }, labelFormat: 'y',
    intervalType: 'Years', valueType: 'DateTime', edgeLabelPlacement:
'Shift'
  };
  const primaryyAxis = {
    title: 'Percentage (%)', labelFormat: '{value}%',
    minimum: 0, maximum: 20, interval: 2
  };
  const titlestyle = { position: 'Bottom', textAlignment: 'Far' };

```



```

    const marker = { visible: true, width: 10, height: 10 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Unemployment Rates 1975-2010'
titleStyle={titlestyle}>
      <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
DateTime]} />
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y'
name='China' width={2} type='StepLine' marker={marker}></SeriesDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y1'
name='Australia' width={2} type='StepLine'
marker={marker}></SeriesDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y2'
name='Japan' width={2} type='StepLine' marker={marker}></SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, FontModel, Legend, DateTime, Tooltip, DataLabel,
StepLineSeries } from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
    { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
    { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
    { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
    { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
    { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
    { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
    { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
  ];
  const primaryxAxis: AxisModel = {
    title: 'Years', lineStyle: { width: 0 }, labelFormat: 'y',
    intervalType: 'Years', valueType: 'DateTime',
edgeLabelPlacement: 'Shift'
  };
  const primaryyAxis: AxisModel = {
    title: 'Percentage (%)', labelFormat: '{value}%',
    minimum: 0, maximum: 20, interval: 2
  };
  const titlestyle: FontModel = { position: 'Bottom', textAlignment:
'Far' };
  const marker = { visible: true, width: 10, height: 10 };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}

```

```

        title='Unemployment Rates 1975-2010'
        titleStyle={titlestyle}>
        <Inject services={[StepLineSeries, Legend, Tooltip, DataLabel,
DateTime]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y'
name='China' width={2} type='StepLine' marker={marker}></SeriesDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y1'
name='Australia' width={2}
                                type='StepLine'
marker={marker}></SeriesDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y2'
name='Japan' width={2} type='StepLine' marker={marker}></SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Title wrap

The `textStyle` property of chart title provides options to customize the size, color, fontFamily, fontWeight, fontStyle, opacity, textAlignment and textOverflow.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
function App() {
    const data = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    const primaryxAxis = { valueType: 'Category', title: 'Countries' };
    const textstyle = { size: '18px', color: 'Red', textAlignment: 'Far',
textOverflow: 'Wrap' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
textStyle={textstyle} title='Sales Analysis'>
        <Inject services={[Legend, Tooltip, DataLabel, Category,
LineSeries]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='month'
yName='sales' type='Line'>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
    }
    ;
    export default App;
}

```

```
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, Legend, Category, Tooltip, DataLabel, LineSeries }
from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    const primaryXAxis: AxisModel = { valueType: 'Category', title:
    'Countries' };
    const textStyle = { size: '18px', color: 'Red', textAlign: 'Far',
    textOverflow: 'Wrap' };
    return <ChartComponent id='charts'
        primaryXAxis={ primaryXAxis }
        textStyle= { textStyle }
        title='Sales Analysis'>
        <Inject services=[Legend, Tooltip, DataLabel, Category,
LineSeries]]/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='month'
yName='sales'
                type='Line'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

Chart subTitle

Chart can be given a subtitle using [subTitle](#) property, to show the information about the data plotted.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, StepLineSeries } from
 '@syncfusion/ej2-react-charts';
function App() {
    const data = [
        { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
        { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
        { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
        { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
        { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
    ];
    return <ChartComponent id='charts'
        primaryXAxis={ primaryXAxis }
        primaryYAxis={ primaryYAxis }
        subtitle='Sales Analysis'
        tooltip= { tooltip }
        legend= { legend }
        dataLabel= { dataLabel }
        seriesCollection=[
            <SeriesDirective dataSource={data} xName='year'
yName='sales'
                type='StepLine'>
            </SeriesDirective>
        ]
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

```

        { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
        { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
        { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
    ];
    const primaryxAxis = { title: 'Years', lineStyle: { width: 0 },
labelFormat: 'y',
        intervalType: 'Years', valueType: 'DateTime', edgeLabelPlacement:
'Shift' };
    const primaryyAxis = { title: 'Percentage (%)', labelFormat: '{value}%',
        minimum: 0, maximum: 20, interval: 2 };
    const subtitlestyle = { fontFamily: "Arial", fontStyle: 'italic',
fontWeight: 'regular',
        color: "#E27F2D" };
    const marker = { visible: true, width: 10, height: 10 };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Unemployment Rates 1975-2010'
subTitle='(1975-2010)' subtitleStyle={subtitlestyle}>
        <Inject services={[StepLineSeries, Legend, Tooltip,
DataLabel, DateTime]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x'
yName='y' name='China' width={2} type='StepLine' marker={marker}>
            </SeriesDirective>
            <SeriesDirective dataSource={data} xName='x'
yName='y1' name='Australia' width={2} type='StepLine' marker={marker}>
            </SeriesDirective>
            <SeriesDirective dataSource={data} xName='x'
yName='y2' name='Japan' width={2} type='StepLine' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { AxisModel, ChartComponent, SeriesCollectionDirective,
SeriesDirective, Inject, FontModel, Legend, DateTime, Tooltip, DataLabel,
StepLineSeries } from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [
        { x: new Date(1975, 0, 1), y: 16, y1: 10, y2: 4.5 },
        { x: new Date(1980, 0, 1), y: 12.5, y1: 7.5, y2: 5 },
        { x: new Date(1985, 0, 1), y: 19, y1: 11, y2: 6.5 },
        { x: new Date(1990, 0, 1), y: 14.4, y1: 7, y2: 4.4 },
        { x: new Date(1995, 0, 1), y: 11.5, y1: 8, y2: 5 },
        { x: new Date(2000, 0, 1), y: 14, y1: 6, y2: 1.5 },
        { x: new Date(2005, 0, 1), y: 10, y1: 3.5, y2: 2.5 },
        { x: new Date(2010, 0, 1), y: 16, y1: 7, y2: 3.7 }
    ];
    const primaryxAxis: AxisModel= {title: 'Years',lineStyle: { width: 0
},labelFormat: 'y',

```

```

        intervalType: 'Years', valueType: 'DateTime', edgeLabelPlacement:
'Shift'} } ;
        const primaryyAxis: AxisModel= {title: 'Percentage (%)', labelFormat:
'{{value}}%',
            minimum: 0, maximum: 20, interval: 2} ;
        const subtitlestyle:FontModel ={fontFamily: "Arial", fontStyle:
'italic',fontWeight: 'regular',
            color: "#E27F2D"};
        const marker={ visible: true, width: 10, height: 10 };
        return <ChartComponent id='charts'
            primaryXAxis={ primaryxAxis }
            primaryYAxis={ primaryyAxis }
            title='Unemployment Rates 1975-2010'
            subTitle='(1975-2010)'
            subTitleStyle = { subtitlestyle }>
            <Inject services={[StepLineSeries, Legend, Tooltip,
DataLabel, DateTime]}/>
            <SeriesCollectionDirective>
                <SeriesDirective dataSource ={data} xName='x'
yName='y' name='China' width={2}
                    type='StepLine' marker={ marker }>
                </SeriesDirective>
                <SeriesDirective dataSource ={data} xName='x'
yName='y1' name='Australia' width={2}
                    type='StepLine' marker={ marker }>
                </SeriesDirective>
                <SeriesDirective dataSource ={data} xName='x'
yName='y2' name='Japan' width={2}
                    type='StepLine' marker={ marker }>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>
    };
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));

```

See also

- [Customize the series points using patterns](#)

<!-- markdownlint-disable MD036 -->

Render methods in React Chart component

Chart uses following two rendering methods.

- SVG
- Canvas

SVG

SVG is used to render Chart by default for all browsers except IE8 and old versions.

Canvas

You can switch between SVG and Canvas rendering by using the `enableCanvas` option. The canvas mode rendering is used in the following scenarios,

- Plotting large number of data points.
- Performing high frequency live updates.

Limitations

- Animation is not supported.

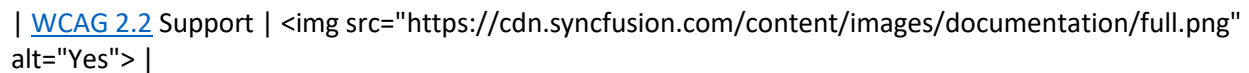
Accessibility in React Chart component

The Chart component followed the accessibility guidelines and standards, including [ADA, Section 508](#), [WCAG 2.2](#) standards, and [WCAG roles](#) that are commonly used to evaluate accessibility.

The accessibility compliance for the Chart component is outlined below.

| Accessibility Criteria | Compatibility |

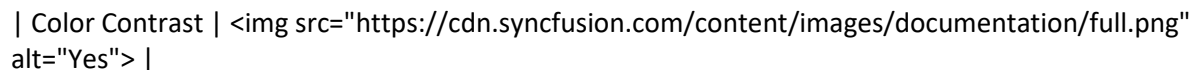
| -- | -- |

| [WCAG 2.2](#) Support |  |

| [Section 508](#) Support |  |

| Screen Reader Support |  |

| Right-To-Left Support |  |

| Color Contrast |  |

| Mobile Device Support |  |

| Keyboard Navigation Support |  |

| [Accessibility Checker](#) Validation |  |

| [Axe-core](#) Accessibility Validation |  |

<style>

```
.post .post-content img {
```

```
display: inline-block;
```

```
margin: 0.5em 0;
```

```
}
```

</style>

<div> - All features of the component meet the requirement.</div>

<div> - Some features of the component do not meet the requirement.</div>

<div> - The component does not meet the requirement.</div>

WAI-ARIA attributes

The Chart component followed the [WAI-ARIA](#) patterns to meet the accessibility. The following ARIA attributes are used in the Chart component:

- img (role)
- button (role)
- region (role)
- aria-label (attribute)
- aria-hidden (attribute)
- aria-pressed (attribute)

Keyboard interaction

The Chart component followed the [keyboard interaction](#) guideline, making it easy for people who use assistive technologies (AT) and those who completely rely on keyboard navigation. The following keyboard shortcuts are supported by the Chart component.

| Press | To do this |

| --- | --- |

| Alt + J | Moves the focus to the chart element. |

| Tab | Moves the focus to the next element in the chart. |

| Shift + Tab | Moves the focus to the previous element in the chart. |

| Down Arrow | Moves the focus to the data point left side from the selected point. |

| Up Arrow | Moves the focus to the data point right side from the selected point. |

| Left Arrow | Moves the focus to the next series in the chart. |

| Right Arrow | Moves the focus to the previous series in the chart. |

| ESC | Cancel the tooltip for the data point. |

| Enter/Space | Selects the data point in the series. |

| Down/Left Arrow | Moves the focus to the legend left side from the selected legend. |

| Up/Right Arrow | Moves the focus to the legend right side from the selected legend. |

| Enter/Space | Toggles the visibility of the corresponding series. |

| Ctrl + + | Zoom in the chart. |

| Ctrl + - | Zoom out the chart. |

| Down/Up Arrow | Pan the chart vertically. |

| Left/Right Arrow | Pan the chart horizontally. |

| R | Reset the zoomed chart. |

| Ctrl + P | Prints the Chart. |

Ensuring accessibility

The Chart component's accessibility levels are ensured through an [accessibility-checker](#) and [axe-core](#) software tools during automated testing.

The accessibility compliance of the Chart component is shown in the following sample. Open the [sample](#) in a new window to evaluate the accessibility of the Chart component with accessibility tools.

See also

- [Accessibility in Syncfusion React components](#)

Internationalization in React Chart component

Chart provide supports for internationalization for below chart elements.

- Datalabel.
- Axis label.
- Tooltip.

For more information about number and date formatter you can refer [internationalization](#).

<!-- markdownlint-disable MD036 -->

Globalization

Globalization is the process of designing and developing an component that works in different cultures/locales. Internationalization library is used to globalize number, date, time values in Chart component using `labelFormat` property in axis.

Numeric Format

In the below example axis, point and tooltip labels are globalized to EUR.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Tooltip, DataLabel } from '@syncfusion/ej2-react-
charts';
import { setCurrencyCode } from '@syncfusion/ej2-base';
setCurrencyCode('EUR');
function App() {
  const data = [
    { x: 1900, y: 4, y1: 2.6, y2: 2.8 }, { x: 1920, y: 3.0, y1: 2.8, y2:
2.5 },
```



```

    { x: 1940, y: 3.8, y1: 2.6, y2: 2.8 }, { x: 1960, y: 3.4, y1: 3, y2:
3.2 },
    { x: 1980, y: 3.2, y1: 3.6, y2: 2.9 }, { x: 2000, y: 3.9, y1: 3, y2:
2 }
  ];
  const primaryxAxis = { edgeLabelPlacement: 'Shift', title: 'Years' };
  const primaryyAxis = { labelFormat: 'c', title: 'Sales Amount in
Millions' };
  const marker = { dataLabel: { visible: true } };
  const tooltip = { enable: true, format: '${point.x} : ${point.y}' };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Average Sales Comparison'
tooltip={tooltip}>
    <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel]}/>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' name='Product
X' type='Column' marker={marker}>
    </SeriesDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y1'
name='Product Y' type='Column' marker={marker}>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Tooltip, DataLabel, TooltipSettingsModel, AxisModel }
from '@syncfusion/ej2-react-charts';
import { loadCldr, setCurrencyCode } from '@syncfusion/ej2-base';
setCurrencyCode('EUR');
function App() {
  const data: any[] = [
    { x: 1900, y: 4, y1: 2.6, y2: 2.8 }, { x: 1920, y: 3.0, y1: 2.8, y2: 2.5
},
    { x: 1940, y: 3.8, y1: 2.6, y2: 2.8 }, { x: 1960, y: 3.4, y1: 3, y2: 3.2
},
    { x: 1980, y: 3.2, y1: 3.6, y2: 2.9 }, { x: 2000, y: 3.9, y1: 3, y2: 2
}
  ];
  const primaryxAxis: AxisModel = { edgeLabelPlacement: 'Shift', title:
'Years' };
  const primaryyAxis: AxisModel = { labelFormat: 'c', title: 'Sales Amount
in Millions' };
  const marker = { dataLabel: { visible: true } };
  const tooltip: TooltipSettingsModel = { enable: true, format: '${point.x}
: ${point.y}' };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}

```

```

        title='Average Sales Comparison'
        tooltip={tooltip}>
        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' name='Product
X' type='Column'
                marker={marker}>
            </SeriesDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y1 '
name='Product Y' type='Column'
                marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Localization in React Chart component

Localization library allows to localize the default text content of Chart. In Chart component, it has the static text on some features(like zooming toolbars) and this can be changed to any other culture(Arabic, Deutsch, French, etc) by defining the locale value and translation object.

<!-- markdownlint-disable MD033 -->

Locale key words	Text to display
Zoom	Zoom
ZoomIn	ZoomIn
ZoomOut	ZoomOut
Reset	Reset
Pan	Pan
ResetZoom	Reset Zoom

To load translation object in an application use load function of L10n class.

For more information about localization, refer this [localization](#)

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Zoom, DataLabel } from '@syncfusion/ej2-react-charts';
import { L10n } from '@syncfusion/ej2-base';
L10n.load({
    'ar-AR': {
        'chart': {
            ZoomIn: 'تكبير',
            ZoomOut: 'تصغير',

```

```

        Zoom: 'زوم',
        Pan: 'مقللة',
        Reset: 'إعادة تعيين',
    },
}
});
function App() {
    const data = [
        { x: 1900, y: 4, y1: 2.6, y2: 2.8 }, { x: 1920, y: 3.0, y1: 2.8, y2: 2.5 },
        { x: 1940, y: 3.8, y1: 2.6, y2: 2.8 }, { x: 1960, y: 3.4, y1: 3, y2: 3.2 },
        { x: 1980, y: 3.2, y1: 3.6, y2: 2.9 }, { x: 2000, y: 3.9, y1: 3, y2: 2 }
    ];
    const primaryxAxis = { edgeLabelPlacement: 'Shift', title: 'Years' };
    const primaryyAxis = { title: 'Sales Amount in Millions' };
    const zoomsettings = {
        enableMouseWheelZooming: true, enablePinchZooming: true,
        enableSelectionZooming: true
    };
    const marker = { dataLabel: { visible: true } };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Average Sales Comparison'
zoomSettings={zoomsettings} locale='ar-AR'>
        <Inject services={[ColumnSeries, Legend, Zoom, DataLabel]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' name='Product
X' type='Column' marker={marker}>
            </SeriesDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y1'
name='Product Y' type='Column' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import {
    ChartComponent, SeriesCollectionDirective, AxesDirective, AxisDirective,
    SeriesDirective, Inject,
    ColumnSeries, Legend, Zoom, DataLabel, ZoomSettingsModel, AxisModel
} from '@syncfusion/ej2-react-charts';
import { L10n } from '@syncfusion/ej2-base';
L10n.load({
    'ar-AR': {
        'chart': {
            ZoomIn: 'تكبير',
            ZoomOut: 'تصغير',
            Zoom: 'زوم',

```

```

        Pan: 'مقلاة',
        Reset: 'إعادة تعيين',
    },
    },
});
function App() {
    const data: any[] = [
        { x: 1900, y: 4, y1: 2.6, y2: 2.8 }, { x: 1920, y: 3.0, y1: 2.8, y2: 2.5 },
    ],
    { x: 1940, y: 3.8, y1: 2.6, y2: 2.8 }, { x: 1960, y: 3.4, y1: 3, y2: 3.2 },
    ],
    { x: 1980, y: 3.2, y1: 3.6, y2: 2.9 }, { x: 2000, y: 3.9, y1: 3, y2: 2 }
    ];
    const primaryxAxis: AxisModel = { edgeLabelPlacement: 'Shift', title: 'Years' };
    const primaryyAxis: AxisModel = { title: 'Sales Amount in Millions' };
    const zoomsettings: ZoomSettingsModel = {
        enableMouseWheelZooming: true, enablePinchZooming: true,
        enableSelectionZooming: true
    };
    const marker = { dataLabel: { visible: true } };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Average Sales Comparison'
        zoomSettings={zoomsettings}
        locale='ar-AR'>
        <Inject services={[ColumnSeries, Legend, Zoom, DataLabel]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' name='Product
X' type='Column'
                marker={marker}>
            </SeriesDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y1'
name='Product Y' type='Column'
                marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Ej1 api migration in React Chart component

This article describes the API migration process of Chart component from Essential JS 1 to Essential JS 2.

Annotations

<!-- markdownlint-disable MD033 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| Annotations | **Property:** *annotations.content*

annotations: [{ content: "watermark"
}],
<EJ.Chart
annotations={annotations}>
</EJ.Chart> | **Property:** *annotations.content*

```

<br/><br/><ChartComponent id='charts'><br/><AnnotationDirective content='<div>Highest
Medal Count</div>'><br/></AnnotationsDirective><br/></ChartComponent>|

|rotation of annotation| Property: annotations <br/><br/>annotations: [{
}],<br/><EJ.Chart<br/>annotations={annotations}> <br/></EJ.Chart>| Not applicable.|

|coordinate unit for annotation| Property: annotations.coordinateUnit <br/><br/>annotations: [{
coordinateUnit : "pixels" }],<br/><EJ.Chart<br/>annotations={annotations}>
<br/></EJ.Chart>| Property: annotations.coordinateUnits <br/><br/><ChartComponent id='charts'>
<br/><AnnotationDirective coordinateUnits='Pixels'>
<br/></AnnotationsDirective><br/></ChartComponent>|

|horizontalAlignment for annotation| Property: annotations.horizontalAlignment
<br/><br/>annotations: [{ horizontalAlignment: "near"
}],<br/><EJ.Chart<br/>annotations={annotations}> <br/></EJ.Chart>| Property:
annotations.horizontalAlignment <br/><br/><ChartComponent id='charts'>
<br/><AnnotationDirective horizontalAlignment='Near'>
<br/></AnnotationsDirective><br/></ChartComponent>|

|horizontalAlignment for annotation| Property: annotations.horizontalAlignment
<br/><br/>annotations: [{ horizontalAlignment: "near"
}],<br/><EJ.Chart<br/>annotations={annotations}> <br/></EJ.Chart>| Property:
annotations.horizontalAlignment <br/><br/><ChartComponent id='charts'>
<br/><AnnotationDirective horizontalAlignment='Near'>
<br/></AnnotationsDirective><br/></ChartComponent>|

|margin for annotation| Property: annotations.margin <br/><br/>annotations: [ margin: { right: 40 }
}],<br/><EJ.Chart<br/>annotations={annotations}> <br/></EJ.Chart>| Not applicable |

|Opacity for annotation| Property: annotations.opacity <br/><br/>annotations: [{ horizontalAlignment:
"near" }],<br/><EJ.Chart<br/>annotations={annotations}> <br/></EJ.Chart>| Property:
annotations.horizontalAlignment <br/><br/><ChartComponent
id='charts'><br/><AnnotationDirective
horizontalAlignment='Near'><br/></AnnotationsDirective><br/></ChartComponent>|

|Region for annotation with respect to chart or series| Property: annotations.region
<br/><br/>annotations: [{ region: "series" }],<br/><EJ.Chart<br/>annotations={annotations}>
<br/></EJ.Chart>| Property: annotations.region <br/><br/><ChartComponent
id='charts'><br/><AnnotationDirective
region='Series'><br/></AnnotationsDirective><br/></ChartComponent>|

|verticalAlignment for annotation| Property: annotations.verticalAlignment <br/><br/>annotations: [{
verticalAlignment: "middle" }],<br/><EJ.Chart<br/>annotations={annotations}>
<br/></EJ.Chart>| Property: annotations.verticalAlignment <br/><br/><ChartComponent
id='charts'><br/><AnnotationDirective
verticalAlignment='Middle'><br/></AnnotationsDirective><br/></ChartComponent>|

|Visibility of annotations| Property: annotations.visible <br/><br/>annotations: [{ visible: true
}],<br/><EJ.Chart<br/>annotations={annotations}> <br/></EJ.Chart>| Not applicable

```

| X offset for annotation | **Property:** *annotations.x*

annotations: [{ x: 170 }],
<EJ.Chart
annotations={annotations}>
</EJ.Chart> | **Property:** *annotations.x*

<ChartComponent id='charts'>
<AnnotationDirective x={170}>
</AnnotationsDirective>
</ChartComponent> |

| X axis name in which annotation to be rendered | **Property:** *annotations.xAxisName*

annotations: [{ xAxisName : "xAxis" }],
<EJ.Chart
annotations={annotations}>
</EJ.Chart> | **Property:** *annotations.xAxisName*

<ChartComponent id='charts'>
<AnnotationDirective xAxisName='xAxis'>
</AnnotationsDirective>
</ChartComponent> |

| Y offset for annotation | **Property:** *annotations.y*

annotations: [{ y: 170 }],
<EJ.Chart
annotations={annotations}>
</EJ.Chart> | **Property:** *annotations.y*

<ChartComponent id='charts'>
<AnnotationDirective y={170}>
</AnnotationsDirective>
</ChartComponent> |

| Y axis name in which annotation to be rendered | **Property:** *annotations.yAxisName*

annotations: [{ yAxisName : "yAxis" }],
<EJ.Chart
annotations={annotations}>
</EJ.Chart> | **Property:** *annotations.yAxisName*

<ChartComponent id='charts'>
<AnnotationDirective yAxisName='yAxis'>
</AnnotationsDirective>
</ChartComponent> |

Columns

<!-- markdownlint-disable MD033 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| Columns in chart | **Property:** *columnType*

var series= [{ type: 'column' }];
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *columnType*

<ChartComponent id='charts'>
<SeriesDirective type='Column'>
</SeriesDirective>
</ChartComponent> |

| width of columns in chart | **Property:** *columnWidth*

var series= [{ columnWidth : 0.8 }];
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *width*

<ChartComponent id='charts' width='80%'>
<SeriesDirective type='Column'>
</SeriesDirective>
</ChartComponent> |

| Rounded column | **Property:** *cornerRadius*

var commonSeriesOptions= { cornerRadius :20 };
<EJ.Chart
commonSeriesOptions={commonSeriesOptions}>
</EJ.Chart> | **Property:** *cornerRadius*

<ChartComponent id='charts' width='80%'>
<SeriesDirective cornerRadius={ bottomLeft: 10 }>
</SeriesDirective>
</ChartComponent> |

CommonSeriesOptions

<!-- markdownlint-disable MD033 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

|commonSeriesOptions| **Property:** *commonSeriesOptions*

var commonSeriesOptions= {
cornerRadius :20 };
<EJ.Chart
commonSeriesOptions={commonSeriesOptions}>

</EJ.Chart>| Not Applicable|

Crosshair

<!-- markdownlint-disable MD033 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

|crosshair| **Property:** *crosshair Label visible*

var crosshair={ visible: true
};
<EJ.Chart
crosshair={crosshair}>
</EJ.Chart>|**Property:** *crosshair enable*

<ChartComponent id='charts' crosshair = {enable:true}>
</ChartComponent>|

|trackballTooltipSettings| **Property:** *trackballTooltipSettings*

var crosshair={ type: 'trackball'
};
<EJ.Chart
crosshair={crosshair}>
</EJ.Chart>| Not applicable

|marker| **Property:** *marker*

var series = [{ marker: { shape: 'Diamond', visible: true
}}];
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:** *marker*

<ChartComponent id='charts'>
<SeriesDirective marker={ visible: true
}}>
</SeriesDirective>
</ChartComponent>|

|crosshair line style| **Property:** *line*

var crosshair={ line: { color: 'gray', width: 2 }
};
<EJ.Chart
crosshair={crosshair}>
</EJ.Chart>|**Property:** *line*

<ChartComponent id='charts' line:{width:2, color:'green'}>
</ChartComponent>|

|type| **Property:** *type*

var crosshair={ type: 'trackball'
};
<EJ.Chart
crosshair={crosshair}>
</EJ.Chart>| Not applicable|

3D chart

<!-- markdownlint-disable MD033 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

|type| **Property:** *type*

var enable3D= true;
<EJ.Chart
enable3D={enable3D}>

</EJ.Chart>| Not applicable|

Indicators

<!-- markdownlint-disable MD033 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

|Type of Indicator| **Property:** *type*

var indicators= [{ type: "accumulationdistribution"
}];
<EJ.Chart
indicators={indicators}>
</EJ.Chart>|**Property:** *marker*

<ChartComponent id='charts'>
<IndicatorDirective
type='AccumulationDistribution'>
</IndicatorDirective>
</ChartComponent>|

|Period for indicator| **Property:** *period*

var indicators= [{ period: 3
}];
<EJ.Chart
indicators={indicators}>
</EJ.Chart>|**Property:** *marker*

```

<br/><br/><ChartComponent id='charts'><br/><IndicatorDirective
period={3}><br/></IndicatorDirective><br/></ChartComponent>|

| %K value in stochastic indicator | Property: kPeriod <br/><br/>var indicators= [{ kPeriod: 3
}];<br/><EJ.Chart<br/>indicators={indicators}> <br/></EJ.Chart>| Property: kPeriod
<br/><br/><ChartComponent id='charts'><br/><IndicatorDirective
kPeriod={2}><br/></IndicatorDirective><br/></ChartComponent>|

| %D value in stochastic indicator | Property: dPeriod <br/><br/>var indicators= [{ dPeriod: 3
}];<br/><EJ.Chart<br/>indicators={indicators}> <br/></EJ.Chart>| Property: dPeriod
<br/><br/><ChartComponent id='charts'><br/><IndicatorDirective
dPeriod={2}><br/></IndicatorDirective><br/></ChartComponent>|

| Shows overSold/overBought values | Not applicable | Property: showZones
<br/><br/><ChartComponent id='charts'><br/><IndicatorDirective
showZones={true}><br/></IndicatorDirective><br/></ChartComponent>|

| Overbought value for RSI and stochastic indicator | Not applicable | Property: overBought
<br/><br/><ChartComponent id='charts'><br/><IndicatorDirective
overBought={70}><br/></IndicatorDirective><br/></ChartComponent>|

| Overbought value for RSI and stochastic indicator | Not applicable | Property: overSold
<br/><br/><ChartComponent id='charts'><br/><IndicatorDirective
overSold={70}><br/></IndicatorDirective><br/></ChartComponent>|

| Field for indicator | Property: field <br/><br/>var indicators= [{ field: "close"
}];<br/><EJ.Chart<br/>indicators={indicators}> <br/></EJ.Chart>| Property: field
<br/><br/><ChartComponent id='charts'><br/><IndicatorDirective
field='Close'><br/></IndicatorDirective><br/></ChartComponent>|

| Slow period for MACD indicator | Property: shortPeriod <br/><br/>var indicators= [{ shortPeriod: 5
}];<br/><EJ.Chart<br/>indicators={indicators}> <br/></EJ.Chart>| Property: slowPeriod
<br/><br/><ChartComponent id='charts'><br/><IndicatorDirective
slowPeriod={5}><br/></IndicatorDirective><br/></ChartComponent>|

| Fast period for MACD indicator | Property: longPeriod <br/><br/>var indicators= [{ longPeriod: 5
}];<br/><EJ.Chart<br/>indicators={indicators}> <br/></EJ.Chart>| Property: fastPeriod
<br/><br/><ChartComponent id='charts'><br/><IndicatorDirective
fastPeriod={5}><br/></IndicatorDirective><br/></ChartComponent>|

| Type of MACD indicator | Property: macdType <br/><br/>var indicators= [{ macdType : "both"
}];<br/><EJ.Chart<br/>indicators={indicators}> <br/></EJ.Chart>| Property: macdType
<br/><br/><ChartComponent id='charts'><br/><IndicatorDirective
macdType='Both'><br/></IndicatorDirective><br/></ChartComponent>|

| Color of the positive bars in Macd indicators | Not applicable | Property: macdPositiveColor
<br/><br/><ChartComponent id='charts'><br/><IndicatorDirective macdPositiveColor:
'red'><br/></IndicatorDirective><br/></ChartComponent>|

```


| Color of the negative bars in Macd indicators | Not applicable | **Property:** *macdPositiveColor*

<ChartComponent id='charts'>
<IndicatorDirective
 macdNegativeColor='#e74c3d'>
</IndicatorDirective>
</ChartComponent>|

| Appearance of upper line in indicator | **Property:** *upperLine*

var indicators= [{ upperLine: {
 fill: '#EECCAA'} }];
<EJ.Chart
indicators={indicators}>
</EJ.Chart>| **Property:** *upperLine*

<ChartComponent id='charts'>
<IndicatorDirective upperLine={ color: '#ffb735',
 width: 1 }>
</IndicatorDirective>
</ChartComponent>|

| Appearance of lower line in indicator | **Property:** *lowerLine*

var indicators= [{ lowerLine: { fill:
 '#EECCAA'} }];
<EJ.Chart
indicators={indicators}>
</EJ.Chart>| **Property:** *lowerLine*

<ChartComponent id='charts'>
<IndicatorDirective lowerLine={ color: '#ffb735',
 width: 1 }>
</IndicatorDirective>
</ChartComponent>|

| Appearance of period line in indicator | **Property:** *periodLine*

var indicators= [{ periodLine: {
 fill: '#EECCAA'} }];
<EJ.Chart
indicators={indicators}>
</EJ.Chart>| **Property:**
periodLine

<ChartComponent id='charts'>
<IndicatorDirective periodLine={ color:
 '#ffb735', width: 1 }>
</IndicatorDirective>
</ChartComponent>|

| Name of the series for which indicator has to be drawn | **Property:** *seriesName*

var
 indicators= [{ seriesName = "Apple Inc" }];
<EJ.Chart
indicators={indicators}>

</EJ.Chart>| **Property:** *seriesName*

<ChartComponent
 id='charts'>
<IndicatorDirective seriesName ='Apple
 Inc'>
</IndicatorDirective>
</ChartComponent>|

| Enabling animation | **Property:** *enableAnimation*

var indicators= [{ enableAnimation : true
 }];
<EJ.Chart
indicators={indicators}>
</EJ.Chart>| **Property:** *animation.enable*

<ChartComponent id='charts'>
<IndicatorDirective animation={ enable: true
 }>
</IndicatorDirective>
</ChartComponent>|

| Animation duration | **Property:** *animationDuration*

var indicators= [{ animationDuration:
 3000 }];
<EJ.Chart
indicators={indicators}>
</EJ.Chart>| **Property:** *animation.duration*

<ChartComponent id='charts'>
<IndicatorDirective animation={ duration:
 3000}>
</IndicatorDirective>
</ChartComponent>|

| Tooltip | **Property:** *tooltip*

var indicators= [{ tooltip: { visible: true }
 }];
<EJ.Chart
indicators={indicators}>
</EJ.Chart>| Not applicable |

| Trigger value of MACD indicator | **Property:** *trigger*

var indicators= [{ trigger: 14
 }];
<EJ.Chart
indicators={indicators}>
</EJ.Chart>| Not applicable |

| Fill color for indicator | **Property:** *fill*

var indicators= [{ fill: '#EEDDCC'
 }];
<EJ.Chart
indicators={indicators}>
</EJ.Chart>| **Property:** *fill*

<ChartComponent id='charts'>
<IndicatorDirective
 fill='#606eff'>
</IndicatorDirective>
</ChartComponent>|

| Width for indicator | **Property:** *width*

var indicators= [{ width: 2
 }];
<EJ.Chart
indicators={indicators}>
</EJ.Chart>| **Property:** *width*

<ChartComponent id='charts'>
<IndicatorDirective
 width={3}>
</IndicatorDirective>
</ChartComponent>|

|xAxis Name of indicator| **Property:** *xAxisName*

var indicators= [{ xAxisName: " }
</EJ.Chart
indicators={indicators}>
</EJ.Chart>|**Property:** *xAxisName*

<ChartComponent id='charts'>
<IndicatorDirective xAxisName="">
</IndicatorDirective>
</ChartComponent>|

|yAxis Name of indicator| **Property:** *yAxisName*

var indicators= [{ yAxisName: " }
</EJ.Chart
indicators={indicators}>
</EJ.Chart>|**Property:** *yAxisName*

<ChartComponent id='charts'>
<IndicatorDirective yAxisName="">
</IndicatorDirective>
</ChartComponent>|

|Visibility of indicator| **Property:** *visibility*

var indicators= [{ visibility: true }
</EJ.Chart
indicators={indicators}>
</EJ.Chart>|Not applicable|

Legend

<!-- markdownlint-disable MD033 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

|Default legend| **Property:** *visible*

var legend= { visible: true }
</EJ.Chart
legend={legend}>
</EJ.Chart>|**Property:** *yAxisName*

<ChartComponent id='charts' legendSettings = { visible: true, position: 'Top' }>
</ChartComponent>|

|Legend height| **Property:** *height*

var legend= { size : { height: 50 } }
</EJ.Chart
legend={legend}>
</EJ.Chart>|**Property:** *yAxisName*

<ChartComponent id='charts' legendSettings = { height: '30' }>
</ChartComponent>|

|Legend width| **Property:** *size.width*

var legend= { size : { width: 50 } }
</EJ.Chart
legend={legend}>
</EJ.Chart>|**Property:** *width*

<ChartComponent id='charts' legendSettings = { width: '30' }>
</ChartComponent>|

|Legend location in chart| **Property:** *location*

var legend= { location: { x: 3, y: 45} }
</EJ.Chart
legend={legend}>
</EJ.Chart>|**Property:** *location*

<ChartComponent id='charts' legendSettings = { location: { x: 3, y: 45} }>
</ChartComponent>|

|Legend position in chart| **Property:** *position*

var legend= { position: 'top' }
</EJ.Chart
legend={legend}>
</EJ.Chart>|**Property:** *position*

<ChartComponent id='charts' legendSettings = { position: 'Top' }>
</ChartComponent>|

|Legend padding| Not applicable| **Property:** *padding*

<ChartComponent id='charts' legendSettings = { padding: 8}>
</ChartComponent>|

|Legend alignment| **Property:** *alignment*

var legend= { alignment: 'center' }
</EJ.Chart
legend={legend}>
</EJ.Chart>|**Property:** *alignment*

<ChartComponent id='charts' legendSettings = { alignment: 'Near' }>
</ChartComponent>|

|text style for legend| **Property:** *font*

var legend= { font: { fontFamily: '' } };
<EJ.Chart
legend={legend}>
</EJ.Chart>| **Property:** *textStyle*

<ChartComponent id='charts' legendSettings = { textStyle: { size: '12px'}}>
</ChartComponent>|

|shape height of legend| **Property:** *itemStyle.height*

var legend= { itemStyle : { height: 20 } };
<EJ.Chart
legend={legend}>
</EJ.Chart>| **Property:** *shapeHeight*

<ChartComponent id='charts' legendSettings = { shapeHeight: 20}>
</ChartComponent>|

|shape width of legend| **Property:** *itemStyle.width*

var legend= { itemStyle : { width: 20 } };
<EJ.Chart
legend={legend}>
</EJ.Chart>| **Property:** *shapeWidth*

<ChartComponent id='charts' legendSettings = { shapeWidth: 20}>
</ChartComponent>|

|shape border of legend| **Property:** *itemStyle.border*

var legend= { itemStyle : { border: {width: 20 } } };
<EJ.Chart
legend={legend}>
</EJ.Chart>| Not Applicable |

|shape padding of legend| **Property:** *itemPadding*

var legend= { itemPadding : 10 };
<EJ.Chart
legend={legend}>
</EJ.Chart>| **Property:** *shapePadding*

<ChartComponent id='charts' legendSettings = { shapePadding: 20}>
</ChartComponent>|

|Background of legend| **Property:** *background*

var legend= { background: 'transparent' };
<EJ.Chart
legend={legend}>
</EJ.Chart>| **Property:** *backgorund*

<ChartComponent id='charts' legendSettings = { background: 'transparent'}>
</ChartComponent>|

|Opacity of legend| **Property:** *opacity*

var legend= { opacity: 0.3 };
<EJ.Chart
legend={legend}>
</EJ.Chart>| **Property:** *opacity*

<ChartComponent id='charts' legendSettings = { opacity: 0.4}>
</ChartComponent>|

|Toggle visibility of series while legend click| **Property:** *toggleSeriesVisibility*

var legend= { toggleSeriesVisibility : true };
<EJ.Chart
legend={legend}>
</EJ.Chart>| **Property:** *toggleVisibility*

<ChartComponent id='charts' legendSettings = { toggleVisibility: true}>
</ChartComponent>|

|Title for legend| **Property:** *title*

var legend= { title: { text: 'LegendTitle' } };
<EJ.Chart
legend={legend}>
</EJ.Chart>| Not applicable |

|Text Overflow for legend| **Property:** *textOverFlow*

var legend= { textOverFlow : 'trim' };
<EJ.Chart
legend={legend}>
</EJ.Chart>| **Property:** *textStyle.textOverFlow*

<ChartComponent id='charts' legendSettings = { text: { textOverFlow : 'trim' } }>
</ChartComponent>|

|Text width for legend while setting text overflow| **Property:** *textWidth*

var legend= { textWidth : 20 };
<EJ.Chart
legend={legend}>
</EJ.Chart>| Not applicable |

|Scroll bar for legend| **Property:** *enableScrollBar*

var legend= { enableScrollBar : true };
<EJ.Chart
legend={legend}>
</EJ.Chart>| Not applicable |

| Row count for legend| **Property:** *rowCount*

var legend= { rowCount : 2
};
<EJ.Chart
legend={legend}>
</EJ.Chart>| Not applicable|

| Column count for legend| **Property:** *columnCount*

var legend= { columnCount :
2};
<EJ.Chart
legend={legend}>
</EJ.Chart>| Not applicable|

| Color for legend items| **Property:** *fill*

var legend= { fill:
'#EEFFCC'};
<EJ.Chart
legend={legend}>
</EJ.Chart>| Not applicable|

primaryXAxis

<!-- markdownlint-disable MD033 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| Axis line cross value| **Property:** *crossesAt*

var primaryXAxis = { crossesAt : 0
};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *crossesAt*

<ChartComponent id='charts' primaryXAxis={ crossesAt : 15
}>
</ChartComponent>|

| axis name with which the axis line has to be crossed| **Property:** *crossesInAxis*

var
primaryXAxis = { crossesInAxis : " };
<EJ.Chart
primaryXAxis={primaryXAxis}>

</EJ.Chart>| **Property:** *crossesInAxis*

<ChartComponent id='charts' primaryXAxis={
crossesInAxis : " }>
</ChartComponent>|

| axis elements placed with axis line| **Property:** *showNextToAxisLine*

var primaryXAxis = {
showNextToAxisLine : true };
<EJ.Chart
primaryXAxis={primaryXAxis}>

</EJ.Chart>| **Property:** *placeNextToAxisLine*

<ChartComponent id='charts'
primaryXAxis={ placeNextToAxisLine: " }>
</ChartComponent>|

| axis line style| **Property:** *axisLine.color*

var primaryXAxis = { axisLine: { color : 'red' }
};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *lineStyle.color*

<ChartComponent id='charts' primaryXAxis={ lineStyle: { color: 'black' }
}>
</ChartComponent>|

| axis line dashArray| **Property:** *axisLine.dashArray*

var primaryXAxis = { axisLine: { dashArray :
'10, 5' } };
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:**
lineStyle.dashArray

<ChartComponent id='charts' primaryXAxis={ lineStyle: {
dashArray: '10, 5' } }>
</ChartComponent>|

| Offset for axis| **Property:** *axisLine.offset*

var primaryXAxis = { axisLine: { offset : 10 }
};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *plotOffset*

<ChartComponent id='charts' primaryXAxis={ plotOffset: 10
}>
</ChartComponent>|

| Visible of an axis| **Property:** *axisLine.visible*

var primaryXAxis = { axisLine: { visible : false }
};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *visible*

<ChartComponent id='charts' primaryXAxis={ visible: false
}>
</ChartComponent>|

| Width of an axis| **Property:** *axisLine.width*

var primaryXAxis = { axisLine: { width : 2 }
};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *lineStyle.width*

```

</></><ChartComponent id='charts' primaryXAxis={ lineStyle: { width: 3
}}></ChartComponent>|

|Column index of an axis| Property: columnIndex </></>var primaryXAxis = { columnIndex: 2
};</><EJ.Chart</>primaryXAxis={primaryXAxis}> </></EJ.Chart>|Property: columnIndex
</></><ChartComponent id='charts' primaryXAxis={ columnIndex:
2}></ChartComponent>|

|span of an axis to place horizontally or vertically| Property: columnSpan </></>var primaryXAxis =
{ columnSpan: 2 };</><EJ.Chart</>primaryXAxis={primaryXAxis}> </></EJ.Chart>|Property:
span </></><ChartComponent id='charts' primaryXAxis={ span:
2}></ChartComponent>|

|Crosshair label of an axis| Property: crossHairLabel.visible </></>var primaryXAxis = {
crossHairLabel: { visible: true } };</><EJ.Chart</>primaryXAxis={primaryXAxis}>
</></EJ.Chart>|Property: crossHairTooltip.enable </></><ChartComponent id='charts'
crossHairTooltip: { enable: true }></ChartComponent>|

|Crosshair label color of an axis| Not applicable |Property: crossHairTooltip.fill
</></><ChartComponent id='charts' crossHairTooltip: { fill: 'red'
}></ChartComponent>|

|Crosshair label text style| Not applicable |Property: crossHairTooltip.textStyle
</></><ChartComponent id='charts' crossHairTooltip: { textStyle: { }
}></ChartComponent>|

|Desired interval count for primaryXAxis| Property: desiredIntervals </></>var primaryXAxis = {
desiredIntervals: 4};</><EJ.Chart</>primaryXAxis={primaryXAxis}> </></EJ.Chart>|Property:
desiredIntervals </></><ChartComponent id='charts' primaryXAxis={ desiredIntervals: 4
}></ChartComponent>|

|Edges primaryXAxis| Property: edgeLabelPlacement </></>var primaryXAxis = {
edgeLabelPlacement: 'none'};</><EJ.Chart</>primaryXAxis={primaryXAxis}>
</></EJ.Chart>|Property: edgeLabelPlacement </></><ChartComponent id='charts'
primaryXAxis={ edgeLabelPlacement: 'Shift' }></ChartComponent>|

|Enables trim for axis labels| Property: enableTrim </></>var primaryXAxis = { enableTrim:
true};</><EJ.Chart</>primaryXAxis={primaryXAxis}> </></EJ.Chart>|Property: enableTrim
</></><ChartComponent id='charts' primaryXAxis={ enableTrim: true
}></ChartComponent>|

|Specifies the interval of the axis according to the zoomed data of the chart| Property:
enableAutoIntervalOnZooming </></>var primaryXAxis = { enableAutoIntervalOnZooming:
true};</><EJ.Chart</>primaryXAxis={primaryXAxis}> </></EJ.Chart>|Property:
enableAutoIntervalOnZooming </></><ChartComponent id='charts' primaryXAxis={
enableAutoIntervalOnZooming: true }></ChartComponent>|

|Font style for primaryXAxis| Property: font </></>var primaryXAxis = { font: { fontFamily: 'Calibri'
};</><EJ.Chart</>primaryXAxis={primaryXAxis}> </></EJ.Chart>|Property: titleStyle
</></><ChartComponent id='charts' primaryXAxis={ titleStyle: {
}}></ChartComponent>|

```

| Indexed for category axis | **Property:** *isIndexed*

var primaryXAxis = { font: { isIndexed: true } };
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart> | **Property:** *isIndexed*

<ChartComponent id='charts' primaryXAxis={ isIndexed: true}>
</ChartComponent> |

| Interval type for date time axis | **Property:** *intervalType*

var primaryXAxis = { intervalType: 'Auto' };
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart> | **Property:** *intervalType*

<ChartComponent id='charts' primaryXAxis={ intervalType: 'Auto'}>
</ChartComponent> |

| Inversed axis | **Property:** *isInversed*

var primaryXAxis = {isInversed: true};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart> | **Property:** *isInversed*

<ChartComponent id='charts' primaryXAxis={ isInversed: true}>
</ChartComponent> |

| Custom label format | **Property:** *labelFormat*

var primaryXAxis = {labelFormat: '{value}K'};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart> | **Property:** *labelFormat*

<ChartComponent id='charts' primaryXAxis={ labelFormat: '{value}K'}>
</ChartComponent> |

| labelIntersectAction | **Property:** *labelIntersectAction*

var primaryXAxis = {labelIntersectAction: 'trim'};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart> | **Property:** *labelIntersectAction*

<ChartComponent id='charts' primaryXAxis={labelIntersectAction: 'Trim' }>
</ChartComponent> |

| labelPosition | **Property:** *labelPosition*

var primaryXAxis = {labelPosition: 'inside'};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart> | **Property:** *labelPosition*

<ChartComponent id='charts' primaryXAxis={labelPosition: 'Inside' }>
</ChartComponent> |

| labelPlacement for category axis | **Property:** *labelPlacement*

var primaryXAxis = {labelPlacement: 'onTicks'};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart> | **Property:** *labelPlacement*

<ChartComponent id='charts' primaryXAxis={ labelPlacement: 'OnTicks' }>
</ChartComponent> |

| Axis label alignment | **Property:** *alignment*

var primaryXAxis = {alignment: 'center'};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart> | Not Applicable |

| Rotation of axis labels | **Property:** *labelRotation*

var primaryXAxis = {labelRotation: 45};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart> | **Property:** *labelRotation*

<ChartComponent id='charts' primaryXAxis={ labelRotation: 45 }>
</ChartComponent> |

| Log base value for logarithmic axis | **Property:** *logBase*

var primaryXAxis = {logBase: 10};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart> | **Property:** *logBase*

<ChartComponent id='charts' primaryXAxis={ logBase: 10 }>
</ChartComponent> |

| Major grid line | **Property:** *majorGridLines.visible*

var primaryXAxis = {majorGridLines: { visible: true}};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart> | Not Applicable |

| Width of MajorGridLines | **Property:** *majorGridLines.width*

var primaryXAxis = {majorGridLines: { width: 2}};
<EJ.Chart
primaryXAxis = {primaryXAxis}>
</EJ.Chart> | **Property:** *majorGridLines.width*

<ChartComponent id='charts' primaryXAxis = { majorGridLines: { width: 2 } }>
</ChartComponent> |

| Color of MajorGridLines | **Property:** *majorGridLines.color*

var primaryXAxis = {majorGridLines: { color: 'black' }};
<EJ.Chart
primaryXAxis = {primaryXAxis}>
</EJ.Chart> | **Property:** *majorGridLines.color*

<ChartComponent id='charts' primaryXAxis = { majorGridLines: { color: 'black' } }>
</ChartComponent> |

| DashArray of MajorGridLines | **Property:** *majorGridLines.dashArray*

var primaryXAxis = {majorGridLines: { dashArray: '10,5' }};
<EJ.Chart
primaryXAxis = {primaryXAxis}>
</EJ.Chart> | **Property:** *majorGridLines.dashArray*

<ChartComponent id='charts' primaryXAxis = { majorGridLines: { dashArray: '10,5' } }>
</ChartComponent> |

| Opacity of major grid line | **Property:** *majorGridLines.opacity*

var primaryXAxis = {majorGridLines: { opacity: true }};
<EJ.Chart
primaryXAxis = {primaryXAxis}>
</EJ.Chart> | Not Applicable |

| Major Tick line | **Property:** *majorTickLines.visible*

var primaryXAxis = {majorTickLines: { visible: true }};
<EJ.Chart
primaryXAxis = {primaryXAxis}>
</EJ.Chart> | Not Applicable |

| Width of MajorTickLines | **Property:** *majorTickLines.width*

var primaryXAxis = {majorTickLines: { width: 2 }};
<EJ.Chart
primaryXAxis = {primaryXAxis}>
</EJ.Chart> | **Property:** *majorTickLines.width*

<ChartComponent id='charts' primaryXAxis = { majorTickLines: { width: 2 } }>
</ChartComponent> |

| Height of MajorTickLines | **Property:** *majorTickLines.size*

var primaryXAxis = {majorTickLines: { size: 2 }};
<EJ.Chart
primaryXAxis = {primaryXAxis}>
</EJ.Chart> | **Property:** *majorTickLines.height*

<ChartComponent id='charts' primaryXAxis = { majorTickLines: { height: 2 } }>
</ChartComponent> |

| Color of MajorTickLines | **Property:** *majorTickLines.color*

var primaryXAxis = {majorTickLines: { color: 'black' }};
<EJ.Chart
primaryXAxis = {primaryXAxis}>
</EJ.Chart> | **Property:** *majorTickLines.color*

<ChartComponent id='charts' primaryXAxis = { majorTickLines: { color: 'black' } }>
</ChartComponent> |

| Opacity of major Tick line | **Property:** *majorTickLines.opacity*

var primaryXAxis = {majorTickLines: { opacity: true}};
<EJ.Chart
primaryXAxis = {primaryXAxis}>
</EJ.Chart> | Not Applicable |

| maximum labels of primaryXAxis | **Property:** *maximumLabels*

var primaryXAxis = { maximumLabels: 5};
<EJ.Chart
primaryXAxis = {primaryXAxis}>
</EJ.Chart> | **Property:** *maximumLabels*

<ChartComponent id='charts' primaryXAxis = { maximumLabels: 4 }>
</ChartComponent> |

| maximum labels width of primaryXAxis to trim | **Property:** *maximumLabelWidth*

var primaryXAxis = { maximumLabelWidth: 40};
<EJ.Chart
primaryXAxis = {primaryXAxis}>
</EJ.Chart> | **Property:** *maximumLabelWidth*

<ChartComponent id='charts' primaryXAxis = { maximumLabelWidth: 40 }>
</ChartComponent> |

|minor grid line| **Property:** *minorGridLines.visible*

var primaryXAxis = { minorGridLines: { visible: true}};
<EJ.Chart
primaryXAxis ={primaryXAxis}>
</EJ.Chart>| Not Applicable|

|Width of minorGridLines| **Property:** *minorGridLines.width*

var primaryXAxis = { minorGridLines: { width: 2}};
<EJ.Chart
primaryXAxis ={primaryXAxis}>
</EJ.Chart>| **Property:** *minorGridLines.width*

<ChartComponent id='charts' primaryXAxis ={ minorGridLines: { width: 2 } }>
</ChartComponent>|

|Color of minorGridLines| **Property:** *minorGridLines.color*

var primaryXAxis = { minorGridLines: { color: 'black'}};
<EJ.Chart
primaryXAxis ={primaryXAxis}>
</EJ.Chart>| **Property:** *minorGridLines.color*

<ChartComponent id='charts' primaryXAxis ={ minorGridLines: { color: 'black' } }>
</ChartComponent>|

|DashArray of minorGridLines| **Property:** *minorGridLines.dashArray*

var primaryXAxis = { minorGridLines: { dashArray: '10,5'}};
<EJ.Chart
primaryXAxis ={primaryXAxis}>
</EJ.Chart>| **Property:** *minorGridLines.dashArray*

<ChartComponent id='charts' primaryXAxis ={ minorGridLines: { dashArray: '10,5' } }>
</ChartComponent>|

|Opacity of minor grid line| **Property:** *minorGridLines.opacity*

var primaryXAxis = { minorGridLines: { opacity: true}};
<EJ.Chart
primaryXAxis ={primaryXAxis}>
</EJ.Chart>| Not Applicable|

|minor Tick line| **Property:** *minorTickLines.visible*

var primaryXAxis = { minorTickLines: { visible: true}};
<EJ.Chart
primaryXAxis ={primaryXAxis}>
</EJ.Chart>| Not Applicable|

|Width of minorTickLines| **Property:** *minorTickLines.width*

var primaryXAxis = { minorTickLines: { width: 2 } };
<EJ.Chart
primaryXAxis ={primaryXAxis}>
</EJ.Chart>| **Property:** *minorTickLines.width*

<ChartComponent id='charts' primaryXAxis ={ minorTickLines: { width: 2}}>
</ChartComponent>|

|Height of minorTickLines| **Property:** *minorTickLines.size*

var primaryXAxis = { minorTickLines: { size: 2}};
<EJ.Chart
primaryXAxis ={primaryXAxis}>
</EJ.Chart>| **Property:** *minorTickLines.height*

<ChartComponent id='charts' primaryXAxis ={ minorTickLines: { height: 2}}>
</ChartComponent>|

|Color of minorTickLines| **Property:** *minorTickLines.color*

var primaryXAxis = { minorTickLines: { color: 'black' } };
<EJ.Chart
primaryXAxis ={primaryXAxis}>
</EJ.Chart>| **Property:** *minorTickLines.color*

<ChartComponent id='charts' primaryXAxis ={ minorTickLines: { color: 'black' } }>
</ChartComponent>|

|Opacity of minor Tick line| **Property:** *minorTickLines.opacity*

var primaryXAxis = { minorTickLines: { opacity: true } };
<EJ.Chart
primaryXAxis ={primaryXAxis}>
</EJ.Chart>| Not Applicable|

|Minor ticks per interval of primaryXAxis| **Property:** *minorTicksPerInterval*

var primaryXAxis = { minorTicksPerInterval: 4};
<EJ.Chart
primaryXAxis ={primaryXAxis}>
</EJ.Chart>| **Property:** *minorTicksPerInterval*

<ChartComponent id='charts' primaryXAxis ={ minorTicksPerInterval: 4 }>
</ChartComponent>|

| name of the primaryXAxis| **Property:** *name*

var primaryXAxis = { name: 'primaryXAxis'};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *name*

<ChartComponent id='charts' primaryXAxis={ name: 'primaryXAxis' }>
</ChartComponent>|

| Orientation of primaryXAxis| **Property:** *orientation*

var primaryXAxis = { orientation: 'Vertical'};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| Not Applicable|

| Plot offset for primaryXAxis| **Property:** *plotOffset*

var primaryXAxis = { plotOffset: 0};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *plotOffset*

<ChartComponent id='charts' primaryXAxis={ plotOffset: 0 }>
</ChartComponent>|

| minimum for primaryXAxis| **Property:** *range.minimum*

var primaryXAxis = { range: { minimum: 10 } };
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *minimum*

<ChartComponent id='charts' primaryXAxis={ minimum: 23 }>
</ChartComponent>|

| maximum for primaryXAxis| **Property:** *range.maximum*

var primaryXAxis = { range: { maximum: 10 } };
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *maximum*

<ChartComponent id='charts' primaryXAxis={ maximum: 23 }>
</ChartComponent>|

| interval for primaryXAxis| **Property:** *range.interval*

var primaryXAxis = { range: { interval: 1 } };
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *interval*

<ChartComponent id='charts' primaryXAxis={ interval: 2 }>
</ChartComponent>|

| RangePadding for primaryXAxis| **Property:** *rangePadding*

var primaryXAxis = { rangePadding: 'None'};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *rangePadding*

<ChartComponent id='charts' primaryXAxis={ rangePadding : 'None' }>
</ChartComponent>|

| Rounding Places in primaryXAxis| **Property:** *roundingPlaces*

var primaryXAxis = { roundingPlaces: 3};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *labelFormat*

<ChartComponent id='charts' primaryXAxis={ labelFormat: 'n3' }>
</ChartComponent>|

| ScrollBar settings of primaryXAxis| **Property:** *scrollbarSettings*

var primaryXAxis = { scrollbarSettings : { } };
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| Not Applicable|

| TickPosition in primaryXAxis| **Property:** *tickLinesPosition*

var primaryXAxis = { tickLinesPosition: 'Inside'};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *tickPosition*

<ChartComponent id='charts' primaryXAxis={ tickPosition: 'Inside' }>
</ChartComponent>|

| valueType of primaryXAxis| **Property:** *valueType*

var primaryXAxis = { valueType: 'DateTime'};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *valueType*

<ChartComponent id='charts' primaryXAxis={ valueType: 'DateTime' }>
</ChartComponent>|

```

|visible of primaryXAxis| Property: visible <br/><br/>var primaryXAxis = { visible:
true};<br/><EJ.Chart<br/>primaryXAxis = {primaryXAxis}> <br/></EJ.Chart>| Property: visible
<br/><br/><ChartComponent id='charts' primaryXAxis = { visible:
true}><br/></ChartComponent>|

|zoomFactor of primaryXAxis| Property: zoomFactor <br/><br/>var primaryXAxis = { zoomFactor:
0.3};<br/><EJ.Chart<br/>primaryXAxis = {primaryXAxis}> <br/></EJ.Chart>| Property: zoomFactor
<br/><br/><ChartComponent id='charts' primaryXAxis = { zoomFactor:
0.3}><br/></ChartComponent>|

|zoomPosition of primaryXAxis| Property: zoomPosition <br/><br/>var primaryXAxis = { zoomPosition:
0.3};<br/><EJ.Chart<br/>primaryXAxis = {primaryXAxis}> <br/></EJ.Chart>| Property: zoomPosition
<br/><br/><ChartComponent id='charts' primaryXAxis = { zoomPosition:
0.3}><br/></ChartComponent>|

|labelBorder of primaryXAxis| Property: labelBorder <br/><br/>var primaryXAxis = { labelBorder: { color:
'red', width: 2}};<br/><EJ.Chart<br/>primaryXAxis = {primaryXAxis}> <br/></EJ.Chart>| Property:
border <br/><br/><ChartComponent id='charts' primaryXAxis = { border: { color: 'red', width: 3
}}><br/></ChartComponent>|

|title of primaryXAxis| Property: title.text <br/><br/>var primaryXAxis = { title: { text: 'Chart title'
}};<br/><EJ.Chart<br/>primaryXAxis = {primaryXAxis}> <br/></EJ.Chart>| Property: title
<br/><br/><ChartComponent id='charts' primaryXAxis = {title: 'Chart
title'}><br/></ChartComponent>|

|StripLine of primaryXAxis| Property: stripLine <br/><br/>var primaryXAxis = { stripLine:
[]};<br/><EJ.Chart<br/>primaryXAxis = {primaryXAxis}> <br/></EJ.Chart>| Property: stripLines
<br/><br/><ChartComponent id='charts' primaryXAxis = {stripLines:
[]}><br/></ChartComponent>|

|Multilevel labels of primaryXAxis| Property: multiLevelLabels <br/><br/>var primaryXAxis = {
multiLevelLabels: []};<br/><EJ.Chart<br/>primaryXAxis = {primaryXAxis}>
<br/></EJ.Chart>| Property: multiLevelLabels <br/><br/><ChartComponent id='charts'
primaryXAxis = {multiLevelLabels: []}><br/></ChartComponent>|

|skeleton for an axes| Not Applicable| Property: axes.skeleton <br/><br/><ChartComponent
id='charts' primaryXAxis = {axes=[ { skeleton: 'yMd' }]}><br/></ChartComponent>|

|skeleton type for an axes| Not Applicable| Property: axes.skeletonType <br/><br/><ChartComponent
id='charts' primaryXAxis = {axes=[{skeletonType: 'DateTime'}]}><br/></ChartComponent>|

primaryYAxis
<!-- markdownlint-disable MD033 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

|Alternate grid band| Property: alternateGridBand <br/><br/>var primaryYAxis = {alternateGridBand: {
even: { fill: 'red' } } };<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}> <br/></EJ.Chart>| Not
applicable|

```

|Axis line cross value| **Property:** *crossesAt*

var primaryYAxis = { crossesAt : 0 };
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>|**Property:** *crossesAt*

<ChartComponent id='charts' primaryYAxis={ crossesAt : 15 }>
</ChartComponent>|

|axis name with which the axis line has to be crossed| **Property:** *crossesInAxis*

var primaryYAxis = { crossesInAxis : " };
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>|**Property:** *crossesInAxis*

<ChartComponent id='charts' primaryYAxis={ crossesInAxis : " }>
</ChartComponent>|

|axis elements placed with axis line| **Property:** *showNextToAxisLine*

var primaryYAxis = { showNextToAxisLine : true };
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>|**Property:** *placeNextToAxisLine*

<ChartComponent id='charts' primaryYAxis={ placeNextToAxisLine: " }>
</ChartComponent>|

|axis line style| **Property:** *axisLine.color*

var primaryYAxis = { axisLine: { color : 'red' } };
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>|**Property:** *lineStyle.color*

<ChartComponent id='charts' primaryYAxis={ lineStyle: { color: 'black' } }>
</ChartComponent>|

|axis line dashArray| **Property:** *axisLine.dashArray*

var primaryYAxis = { axisLine: { dashArray : '10, 5' } };
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>|**Property:** *lineStyle.dashArray*

<ChartComponent id='charts' primaryYAxis={ lineStyle: { dashArray: '10, 5' } }>
</ChartComponent>|

|Offset for axis| **Property:** *axisLine.offset*

var primaryYAxis = { axisLine: { offset : 10 } };
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>|**Property:** *plotOffset*

<ChartComponent id='charts' primaryYAxis={ plotOffset: 10 }>
</ChartComponent>|

|Visible of an axis| **Property:** *axisLine.visible*

var primaryYAxis = { axisLine: { visible : false } };
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>|**Property:** *visible*

<ChartComponent id='charts' primaryYAxis={ visible: false }>
</ChartComponent>|

|Width of an axis| **Property:** *axisLine.width*

var primaryYAxis = { axisLine: { width : 2 } };
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>|**Property:** *lineStyle.width*

<ChartComponent id='charts' primaryYAxis={ lineStyle: { width: 3 } }>
</ChartComponent>|

|Column index of an axis| **Property:** *columnIndex*

var primaryYAxis = { columnIndex: 2 };
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>|**Property:** *columnIndex*

<ChartComponent id='charts' primaryYAxis={ columnIndex: 2 }>
</ChartComponent>|

|span of an axis to place horizontally or vertically| **Property:** *columnSpan*

var primaryYAxis = { columnSpan: 2 };
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>|**Property:** *span*

<ChartComponent id='charts' primaryYAxis={ span: 2 }>
</ChartComponent>|

| Crosshair label of an axis| **Property:** *crossHairLabel.visible*

var primaryYAxis = {
crossHairLabel: { visible: true } };
<EJ.Chart
primaryYAxis = {primaryYAxis}>

</EJ.Chart>| **Property:** *crossHairTooltip.enable*

<ChartComponent id='charts'
crossHairTooltip: { enable: true }>
</ChartComponent>|

| Crosshair label color of an axis| Not applicable | **Property:** *crossHairTooltip.fill*

<ChartComponent id='charts' crossHairTooltip: { fill: 'red'
>
</ChartComponent>|

| Crosshair label text style| Not applicable | **Property:** *crossHairTooltip.textStyle*

<ChartComponent id='charts' crossHairTooltip: { textStyle: { }
>
</ChartComponent>|

| Desired interval count for primaryYAxis| **Property:** *desiredIntervals*

var primaryYAxis = {
desiredIntervals: 4};
<EJ.Chart
primaryYAxis = {primaryYAxis}>
</EJ.Chart>| **Property:**
desiredIntervals

<ChartComponent id='charts' primaryYAxis = { desiredIntervals: 4
>
</ChartComponent>|

| Edges primaryYAxis| **Property:** *edgeLabelPlacement*

var primaryYAxis = {
edgeLabelPlacement: 'none'};
<EJ.Chart
primaryYAxis = {primaryYAxis}>

</EJ.Chart>| **Property:** *edgeLabelPlacement*

<ChartComponent id='charts'
primaryYAxis = { edgeLabelPlacement: 'Shift' }>
</ChartComponent>|

| Enables trim for axis labels| **Property:** *enableTrim*

var primaryYAxis = { enableTrim:
true};
<EJ.Chart
primaryYAxis = {primaryYAxis}>
</EJ.Chart>| **Property:** *enableTrim*

<ChartComponent id='charts' primaryYAxis = { enableTrim: true
>
</ChartComponent>|

| Specifies the interval of the axis according to the zoomed data of the chart| **Property:**
enableAutoIntervalOnZooming

var primaryYAxis = { enableAutoIntervalOnZooming:
true};
<EJ.Chart
primaryYAxis = {primaryYAxis}>
</EJ.Chart>| **Property:**
enableAutoIntervalOnZooming

<ChartComponent id='charts' primaryYAxis = {
enableAutoIntervalOnZooming: true }>
</ChartComponent>|

| Font style for primaryYAxis| **Property:** *font*

var primaryYAxis = { font: { fontFamily: 'Calibri'
};
<EJ.Chart
primaryYAxis = {primaryYAxis}>
</EJ.Chart>| **Property:** *titleStyle*

<ChartComponent id='charts' primaryYAxis = { titleStyle: {
}}>
</ChartComponent>|

| Indexed for category axis| **Property:** *isIndexed*

var primaryYAxis = { font: { isIndexed: true}
};
<EJ.Chart
primaryYAxis = {primaryYAxis}>
</EJ.Chart>| **Property:** *isIndexed*

<ChartComponent id='charts' primaryYAxis = { isIndexed:
true}>
</ChartComponent>|

| Interval type for date time axis| **Property:** *intervalType*

var primaryYAxis = { intervalType:
'Auto' };
<EJ.Chart
primaryYAxis = {primaryYAxis}>
</EJ.Chart>| **Property:**
intervalType

<ChartComponent id='charts' primaryYAxis = { intervalType:
'Auto' }>
</ChartComponent>|

| Inversed axis| **Property:** *isInversed*

var primaryYAxis = { isInversed:
true};
<EJ.Chart
primaryYAxis = {primaryYAxis}>
</EJ.Chart>| **Property:** *isInversed*

```
<br/><br/><ChartComponent id='charts' primaryYAxis = { isInversed:
true}><br/></ChartComponent>|
```

```
| Custom label format | Property: labelFormat <br/><br/>var primaryYAxis = {labelFormat:
'{value}K'};<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}> <br/></EJ.Chart>| Property:
labelFormat <br/><br/><ChartComponent id='charts' primaryYAxis = { labelFormat:
'{value}K'}><br/></ChartComponent>|
```

```
| labelIntersectAction | Property: labelIntersectAction <br/><br/>var primaryYAxis =
{labelIntersectAction: 'trim'};<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}>
<br/></EJ.Chart>| Property: labelIntersectAction <br/><br/><ChartComponent id='charts'
primaryYAxis = {labelIntersectAction: 'Trim' }><br/></ChartComponent>|
```

```
| labelPosition | Property: labelPosition <br/><br/>var primaryYAxis = {labelPosition:
'inside'};<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}> <br/></EJ.Chart>| Property:
labelPosition <br/><br/><ChartComponent id='charts' primaryYAxis = {labelPosition: 'Inside'
}><br/></ChartComponent>|
```

```
| labelPlacement for category axis | Property: labelPlacement <br/><br/>var primaryYAxis =
{labelPlacement: 'onTicks'};<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}>
<br/></EJ.Chart>| Property: labelPlacement <br/><br/><ChartComponent id='charts' primaryYAxis
= { labelPlacement: 'OnTicks' }><br/></ChartComponent>|
```

```
| Axis label alignment | Property: alignment <br/><br/>var primaryYAxis = {alignment:
'center'};<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}> <br/></EJ.Chart>| Not Applicable |
```

```
| Rotation of axis labels | Property: labelRotation <br/><br/>var primaryYAxis = {labelRotation:
45};<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}> <br/></EJ.Chart>| Property: labelRotation
<br/><br/><ChartComponent id='charts' primaryYAxis = { labelRotation: 45
}><br/></ChartComponent>|
```

```
| Log base value for logarithmic axis | Property: logBase <br/><br/>var primaryYAxis = {logBase:
10};<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}> <br/></EJ.Chart>| Property: logBase
<br/><br/><ChartComponent id='charts' primaryYAxis = { logBase: 10
}><br/></ChartComponent>|
```

```
| Major grid line | Property: majorGridLines.visible <br/><br/>var primaryYAxis = {majorGridLines: {
visible: true}};<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}> <br/></EJ.Chart>| Not Applicable |
```

```
| Width of MajorGridLines | Property: majorGridLines.width <br/><br/>var primaryYAxis =
{majorGridLines: { width: 2}};<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}>
<br/></EJ.Chart>| Property: majorGridLines.width <br/><br/><ChartComponent id='charts'
primaryYAxis = { majorGridLines: { width: 2 } }><br/></ChartComponent>|
```

```
| Color of MajorGridLines | Property: majorGridLines.color <br/><br/>var primaryYAxis =
{majorGridLines: { color: 'black' }};<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}>
<br/></EJ.Chart>| Property: majorGridLines.color <br/><br/><ChartComponent id='charts'
primaryYAxis = { majorGridLines: { color: 'black' } }><br/></ChartComponent>|
```

```
| DashArray of MajorGridLines | Property: majorGridLines.dashArray <br/><br/>var primaryYAxis =
{majorGridLines: { dashArray: '10,5' }};<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}>
```

`
</EJ.Chart>` | **Property:** `majorGridLines.dashArray` `

<ChartComponent id='charts'`
`primaryYAxis = { majorGridLines: { dashArray: '10,5' } }>
</ChartComponent>` |

| Opacity of major grid line | **Property:** `majorGridLines.opacity` `

var primaryYAxis =`
`{majorGridLines: { opacity: true }};
<EJ.Chart
primaryYAxis = {primaryYAxis}>`
`
</EJ.Chart>` | Not Applicable |

| Major Tick line | **Property:** `majorTickLines.visible` `

var primaryYAxis = {majorTickLines: {`
`visible: true }};
<EJ.Chart
primaryYAxis = {primaryYAxis}>` `
</EJ.Chart>` | Not
 Applicable |

| Width of MajorTickLines | **Property:** `majorTickLines.width` `

var primaryYAxis =`
`{majorTickLines: { width: 2 }};
<EJ.Chart
primaryYAxis = {primaryYAxis}>`
`
</EJ.Chart>` | **Property:** `majorTickLines.width` `

<ChartComponent id='charts'`
`primaryYAxis = { majorTickLines: { width: 2 } }>
</ChartComponent>` |

| Height of MajorTickLines | **Property:** `majorTickLines.size` `

var primaryYAxis = {majorTickLines:`
`{ size: 2 }};
<EJ.Chart
primaryYAxis = {primaryYAxis}>` `
</EJ.Chart>` | **Property:**
`majorTickLines.height` `

<ChartComponent id='charts' primaryYAxis = { majorTickLines: {`
`height: 2 } }>
</ChartComponent>` |

| Color of MajorTickLines | **Property:** `majorTickLines.color` `

var primaryYAxis = {majorTickLines:`
`{ color: 'black' }};
<EJ.Chart
primaryYAxis = {primaryYAxis}>` `
</EJ.Chart>` | **Property:**
`majorTickLines.color` `

<ChartComponent id='charts' primaryYAxis = { majorTickLines: {`
`color: 'black' } }>
</ChartComponent>` |

| Opacity of major Tick line | **Property:** `majorTickLines.opacity` `

var primaryYAxis =`
`{majorTickLines: { opacity: true }};
<EJ.Chart
primaryYAxis = {primaryYAxis}>`
`
</EJ.Chart>` | Not Applicable |

| maximum labels of primaryYAxis | **Property:** `maximumLabels` `

var primaryYAxis = {`
`maximumLabels: 5}};
<EJ.Chart
primaryYAxis = {primaryYAxis}>` `
</EJ.Chart>` | **Property:**
`maximumLabels` `

<ChartComponent id='charts' primaryYAxis = { maximumLabels: 4`
`}>
</ChartComponent>` |

| maximum labels width of primaryYAxis to trim | **Property:** `maximumLabelWidth` `

var`
`primaryYAxis = { maximumLabelWidth: 40}};
<EJ.Chart
primaryYAxis = {primaryYAxis}>`
`
</EJ.Chart>` | **Property:** `maximumLabelWidth` `

<ChartComponent id='charts'`
`primaryYAxis = { maximumLabelWidth: 40 }>
</ChartComponent>` |

| minor grid line | **Property:** `minorGridLines.visible` `

var primaryYAxis = { minorGridLines: {`
`visible: true }};
<EJ.Chart
primaryYAxis = {primaryYAxis}>` `
</EJ.Chart>` | Not
 Applicable |

| Width of minorGridLines | **Property:** `minorGridLines.width` `

var primaryYAxis = {`
`minorGridLines: { width: 2 }};
<EJ.Chart
primaryYAxis = {primaryYAxis}>`
`
</EJ.Chart>` | **Property:** `minorGridLines.width` `

<ChartComponent id='charts'`
`primaryYAxis = { minorGridLines: { width: 2 } }>
</ChartComponent>` |

| Color of minorGridLines | **Property:** `minorGridLines.color` `

var primaryYAxis = {`
`minorGridLines: { color: 'black'}};
<EJ.Chart
primaryYAxis = {primaryYAxis}>`

```

<br/></EJ.Chart> | Property: minorGridLines.color <br/><br/><ChartComponent id='charts'
primaryYAxis = { minorGridLines: { color: 'black' } }><br/></ChartComponent> |
| DashArray of minorGridLines | Property: minorGridLines.dashArray <br/><br/>var primaryYAxis = {
minorGridLines: { dashArray: '10,5' } };<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}>
<br/></EJ.Chart> | Property: minorGridLines.dashArray <br/><br/><ChartComponent id='charts'
primaryYAxis = { minorGridLines: { dashArray: '10,5' } }><br/></ChartComponent> |
| Opacity of minor grid line | Property: minorGridLines.opacity <br/><br/>var primaryYAxis = {
minorGridLines: { opacity: true } };<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}>
<br/></EJ.Chart> | Not Applicable |
| minor Tick line | Property: minorTickLines.visible <br/><br/>var primaryYAxis = { minorTickLines: {
visible: true } };<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}> <br/></EJ.Chart> | Not Applicable |
| Width of minorTickLines | Property: minorTickLines.width <br/><br/>var primaryYAxis = {
minorTickLines: { width: 2 } };<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}>
<br/></EJ.Chart> | Property: minorTickLines.width <br/><br/><ChartComponent id='charts'
primaryYAxis = { minorTickLines: { width: 2 } }><br/></ChartComponent> |
| Height of minorTickLines | Property: minorTickLines.size <br/><br/>var primaryYAxis = { minorTickLines:
{ size: 2 } };<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}> <br/></EJ.Chart> | Property:
minorTickLines.height <br/><br/><ChartComponent id='charts' primaryYAxis = { minorTickLines: {
height: 2 } }><br/></ChartComponent> |
| Color of minorTickLines | Property: minorTickLines.color <br/><br/>var primaryYAxis = { minorTickLines:
{ color: 'black' } };<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}> <br/></EJ.Chart> | Property:
minorTickLines.color <br/><br/><ChartComponent id='charts' primaryYAxis = { minorTickLines: {
color: 'black' } }><br/></ChartComponent> |
| Opacity of minor Tick line | Property: minorTickLines.opacity <br/><br/>var primaryYAxis = {
minorTickLines: { opacity: true } };<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}>
<br/></EJ.Chart> | Not Applicable |
| Minor ticks per interval of primaryYAxis | Property: minorTicksPerInterval <br/><br/>var primaryYAxis =
{ minorTicksPerInterval: 4 };<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}>
<br/></EJ.Chart> | Property: minorTicksPerInterval <br/><br/><ChartComponent id='charts'
primaryYAxis = { minorTicksPerInterval: 4 } }><br/></ChartComponent> |
| name of the primaryYAxis | Property: name <br/><br/>var primaryYAxis = { name:
'primaryYAxis' };<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}> <br/></EJ.Chart> | Property:
name <br/><br/><ChartComponent id='charts' primaryYAxis = { name: 'primaryYAxis'
}><br/></ChartComponent> |
| Orientation of primaryYAxis | Property: orientation <br/><br/>var primaryYAxis = { orientation:
'Vertical' };<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}> <br/></EJ.Chart> | Not Applicable |
| Plot offset for primaryYAxis | Property: plotOffset <br/><br/>var primaryYAxis = { plotOffset:
0 };<br/><EJ.Chart<br/>primaryYAxis = {primaryYAxis}> <br/></EJ.Chart> | Property: plotOffset
<br/><br/><ChartComponent id='charts' primaryYAxis = { plotOffset: 0
}><br/></ChartComponent> |

```

|minimum for primaryYAxis| **Property:** *range.minimum*

var primaryYAxis = { range: { minimum: 10 }};
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>| **Property:** *minimum*

<ChartComponent id='charts' primaryYAxis={ minimum: 23 }>
</ChartComponent>|

|maximum for primaryYAxis| **Property:** *range.maximum*

var primaryYAxis = { range: { maximum: 10 }};
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>| **Property:** *maximum*

<ChartComponent id='charts' primaryYAxis={ maximum: 23 }>
</ChartComponent>|

|interval for primaryYAxis| **Property:** *range.interval*

var primaryYAxis = { range: { interval: 1 }};
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>| **Property:** *interval*

<ChartComponent id='charts' primaryYAxis={ interval: 2 }>
</ChartComponent>|

|RangePadding for primaryYAxis| **Property:** *rangePadding*

var primaryYAxis = { rangePadding: 'None'};
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>| **Property:** *rangePadding*

<ChartComponent id='charts' primaryYAxis={ rangePadding: 'None'}>
</ChartComponent>|

|Rounding Places in primaryYAxis| **Property:** *roundingPlaces*

var primaryYAxis = { roundingPlaces: 3};
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>| **Property:** *labelFormat*

<ChartComponent id='charts' primaryYAxis={ labelFormat: 'n3'}>
</ChartComponent>|

|ScrollBar settings of primaryYAxis| **Property:** *scrollbarSettings*

var primaryYAxis = { scrollbarSettings : { }};
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>| Not Applicable|

|TickPosition in primaryYAxis| **Property:** *tickLinesPosition*

var primaryYAxis = { tickLinesPosition: 'Inside'};
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>| **Property:** *tickPosition*

<ChartComponent id='charts' primaryYAxis={ tickPosition: 'Inside'}>
</ChartComponent>|

|valueType of primaryYAxis| **Property:** *valueType*

var primaryYAxis = { valueType: 'DateTime'};
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>| **Property:** *valueType*

<ChartComponent id='charts' primaryYAxis={ valueType: 'DateTime'}>
</ChartComponent>|

|visible of primaryYAxis| **Property:** *visible*

var primaryYAxis = { visible: true};
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>| **Property:** *visible*

<ChartComponent id='charts' primaryYAxis={ visible: true}>
</ChartComponent>|

|zoomFactor of primaryYAxis| **Property:** *zoomFactor*

var primaryYAxis = { zoomFactor: 0.3};
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>| **Property:** *zoomFactor*

<ChartComponent id='charts' primaryYAxis={ zoomFactor: 0.3}>
</ChartComponent>|

|zoomPosition of primaryYAxis| **Property:** *zoomPosition*

var primaryYAxis = { zoomPosition: 0.3};
<EJ.Chart
primaryYAxis={primaryYAxis}>
</EJ.Chart>| **Property:** *zoomPosition*


```

<br/><br/><ChartComponent id='charts' primaryYAxis={ zoomPosition:
0.3}><br/></ChartComponent>|

|labelBorder of primaryYAxis| Property: labelBorder <br/><br/>var primaryYAxis = { labelBorder: { color:
'red', width: 2}};<br/><EJ.Chart<br/>primaryYAxis={primaryYAxis}> <br/></EJ.Chart>|Property:
border <br/><br/><ChartComponent id='charts' primaryYAxis={ border: { color: 'red', width: 3
}}><br/></ChartComponent>|

|title of primaryYAxis| Property: title.text <br/><br/>var primaryYAxis = { title: { text: 'Chart title'
}};<br/><EJ.Chart<br/>primaryYAxis={primaryYAxis}> <br/></EJ.Chart>|Property: title
<br/><br/><ChartComponent id='charts' primaryYAxis={title: 'Chart
title'}><br/></ChartComponent>|

|StripLine of primaryYAxis| Property: stripline <br/><br/>var primaryYAxis = { stripline:
[]}<br/><EJ.Chart<br/>primaryYAxis={primaryYAxis}> <br/></EJ.Chart>|Property: stripLines
<br/><br/><ChartComponent id='charts' primaryYAxis={stripLines:
[]}><br/></ChartComponent>|

|Multilevel labels of primaryYAxis| Property: multiLevelLabels <br/><br/>var primaryYAxis = {
multiLevelLabels: []}<br/><EJ.Chart<br/>primaryYAxis={primaryYAxis}>
<br/></EJ.Chart>|Property: multiLevelLabels <br/><br/><ChartComponent id='charts'
primaryYAxis={multiLevelLabels: []}><br/></ChartComponent>|

|skeleton for an axes| Not Applicable|Property: axes.skeleton <br/><br/><ChartComponent
id='charts' primaryYAxis={axes=[ { skeleton: 'yMd' }]}><br/></ChartComponent>|

|skeleton type for an axes| Not Applicable|Property: axes.skeletonType <br/><br/><ChartComponent
id='charts' primaryYAxis={axes=[{skeletonType: 'DateTime'}]}><br/></ChartComponent>|

Axes
<!-- markdownlint-disable MD033 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

|Alternate grid band| Property: alternateGridBand <br/><br/>var axes= {alternateGridBand: { even: {
fill: 'red' } } };<br/><EJ.Chart<br/>axes={axes}> <br/></EJ.Chart>|Not applicable|

|Axis line cross value| Property: crossesAt <br/><br/>var axes= { crossesAt : 0
}<br/><EJ.Chart<br/>axes={axes}> <br/></EJ.Chart>|Property: crossesAt
<br/><br/><ChartComponent id='charts'><br/><AxisDirective
crossesAt='4'><br/></AxisDirective><br/></ChartComponent>|

|axis name with which the axis line has to be crossed| Property: crossesInAxis <br/><br/>var axes= {
crossesInAxis : '' };<br/><EJ.Chart<br/>axes={axes}> <br/></EJ.Chart>|Property: crossesInAxis
<br/><br/><ChartComponent id='charts'><br/><AxisDirective crossesInAxis : ""
><br/></AxisDirective><br/></ChartComponent>|

|axis elements placed with axis line| Property: showNextToAxisLine <br/><br/>var axes= {
showNextToAxisLine : true };<br/><EJ.Chart<br/>axes={axes}> <br/></EJ.Chart>|Property:

```

```

placeNextToAxisLine <br/><br/><ChartComponent id='charts'><br/><AxisDirective
placeNextToAxisLine: ""><br/></AxisDirective><br/></ChartComponent>|

|axis line style| Property: axisLine.color <br/><br/>var axes= { axisLine: { color : 'red' }
};<br/><EJ.Chart<br/>axes={axes}> <br/></EJ.Chart>| Property: lineStyle.color
<br/><br/><ChartComponent id='charts'><br/><AxisDirective lineStyle: { color: 'black'
}><br/></AxisDirective><br/></ChartComponent>|

|axis line dashArray| Property: axisLine.dashArray <br/><br/>var axes= { axisLine: { dashArray : '10, 5' }
};<br/><EJ.Chart<br/>axes={axes}> <br/></EJ.Chart>| Property: lineStyle.dashArray
<br/><br/><ChartComponent id='charts'><br/><AxisDirective lineStyle: { dashArray: '10, 5'
}><br/></AxisDirective><br/></ChartComponent>|

|Offset for axis| Property: axisLine.offset <br/><br/>var axes= { axisLine: { offset : 10 }
};<br/><EJ.Chart<br/>axes={axes}> <br/></EJ.Chart>| Property: plotOffset
<br/><br/><ChartComponent id='charts'><br/><AxisDirective plotOffset:
10><br/></AxisDirective><br/></ChartComponent>|

|Visible of an axis| Property: axisLine.visible <br/><br/>var axes= { axisLine: { visible : false }
};<br/><EJ.Chart<br/>axes={axes}> <br/></EJ.Chart>| Property: visible <br/><br/><ChartComponent
id='charts'><br/><AxisDirective visible: false><br/></AxisDirective><br/></ChartComponent>|

|Width of an axis| Property: axisLine.width <br/><br/>var axes= { axisLine: { width : 2 }
};<br/><EJ.Chart<br/>axes={axes}> <br/></EJ.Chart>| Property: lineStyle.width
<br/><br/><ChartComponent id='charts'><br/><AxisDirective lineStyle: { width: 3
}><br/></AxisDirective><br/></ChartComponent>|

|Column index of an axis| Property: columnIndex <br/><br/>var axes= { columnIndex: 2
};<br/><EJ.Chart<br/>axes={axes}> <br/></EJ.Chart>| Property: columnIndex
<br/><br/><ChartComponent id='charts'><br/><AxisDirective columnIndex:
2><br/></AxisDirective><br/></ChartComponent>|

|span of an axis to place horizontally or vertically| Property: columnSpan <br/><br/>var axes= {
columnSpan: 2 };<br/><EJ.Chart<br/>axes={axes}> <br/></EJ.Chart>| Property: span
<br/><br/><ChartComponent id='charts'><br/><AxisDirective span:
2><br/></AxisDirective><br/></ChartComponent>|

|Crosshair label of an axis| Property: crossHairLabel.visible <br/><br/>var axes= { crossHairLabel: {
visible: true } };<br/><EJ.Chart<br/>axes={axes}> <br/></EJ.Chart>| Property: crossHairTooltip.enable
<br/><br/><ChartComponent id='charts'><br/><AxisDirective crossHairTooltip: { enable: true
}><br/></AxisDirective><br/></ChartComponent>|

|Crosshair label color of an axis| Not applicable | Property: crossHairTooltip.fill
<br/><br/><ChartComponent id='charts'><br/><AxisDirective crossHairTooltip: { fill: 'red' }
><br/></AxisDirective><br/></ChartComponent>|

|Crosshair label text style| Not applicable | Property: crossHairTooltip.textStyle
<br/><br/><ChartComponent id='charts'><br/><AxisDirective crossHairTooltip: { textStyle: { }
}><br/></AxisDirective><br/></ChartComponent>|

```

| Desired interval count for primaryYAxis | **Property:** *desiredIntervals*

var axes= {
desiredIntervals: 4};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *desiredIntervals*

<ChartComponent id='charts'>
<AxisDirective desiredIntervals:
4>
</AxisDirective>
</ChartComponent> |

| Edges axes | **Property:** *edgeLabelPlacement*

var axes= { edgeLabelPlacement:
'none'};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *edgeLabelPlacement*

<ChartComponent id='charts'>
<AxisDirective edgeLabelPlacement:
'Shift'>
</AxisDirective>
</ChartComponent> |

| Enables trim for axis labels | **Property:** *enableTrim*

var axes= { enableTrim:
true};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *enableTrim*

<ChartComponent id='charts'>
<AxisDirective enableTrim:
true>
</AxisDirective>
</ChartComponent> |

| Specifies the interval of the axis according to the zoomed data of the chart | **Property:**
enableAutoIntervalOnZooming

var axes= { enableAutoIntervalOnZooming:
true};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *enableAutoIntervalOnZooming*

<ChartComponent id='charts'>
<AxisDirective enableAutoIntervalOnZooming:
true>
</AxisDirective>
</ChartComponent> |

| Font style for axes | **Property:** *font*

var axes= { font: { fontFamily: 'Calibri'}
};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *titleStyle*

<ChartComponent id='charts'>
<AxisDirective titleStyle: {
}>
</AxisDirective>
</ChartComponent> |

| Indexed for category axis | **Property:** *isIndexed*

var axes= { font: { isIndexed: true}
};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *isIndexed*

<ChartComponent id='charts'>
<AxisDirective isIndexed:
true>
</AxisDirective>
</ChartComponent> |

| Interval type for date time axis | **Property:** *intervalType*

var axes= { intervalType: 'Auto'
};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *intervalType*

<ChartComponent id='charts'>
<AxisDirective intervalType:
'Auto'>
</AxisDirective>
</ChartComponent> |

| Inversed axis | **Property:** *isInversed*

var axes= {isInversed:
true};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *isInversed*

<ChartComponent id='charts'>
<AxisDirective isInversed:
true>
</AxisDirective>
</ChartComponent> |

| Custom label format | **Property:** *labelFormat*

var axes= {labelFormat:
'{value}K'};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *labelFormat*

<ChartComponent id='charts'>
<AxisDirective labelFormat:
'{value}K'>
</AxisDirective>
</ChartComponent> |

| labelIntersectAction | **Property:** *labelIntersectAction*

var axes= {labelIntersectAction:
'trim'};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *labelIntersectAction*

<ChartComponent id='charts'>
<AxisDirective labelIntersectAction:
'Trim'>
</AxisDirective>
</ChartComponent> |

|labelPosition| **Property:** *labelPosition*

var axes= {labelPosition: 'inside'};
<EJ.Chart
axes={axes}>
</EJ.Chart>| **Property:** *labelPosition*

<ChartComponent id='charts'>
<AxisDirective labelPosition: 'Inside'>
</AxisDirective>
</ChartComponent>|

|labelPlacement for category axis| **Property:** *labelPlacement*

var axes= {labelPlacement: 'onTicks'};
<EJ.Chart
axes={axes}>
</EJ.Chart>| **Property:** *labelPlacement*

<ChartComponent id='charts'>
<AxisDirective labelPlacement: 'OnTicks'>
</AxisDirective>
</ChartComponent>|

|Axis label alignment| **Property:** *alignment*

var axes= {alignment: 'center'};
<EJ.Chart
axes={axes}>
</EJ.Chart>| Not Applicable |

|Rotation of axis labels| **Property:** *labelRotation*

var axes= {labelRotation: 45};
<EJ.Chart
axes={axes}>
</EJ.Chart>| **Property:** *labelRotation*

<ChartComponent id='charts'>
<AxisDirective labelRotation: 45>
</AxisDirective>
</ChartComponent>|

|Log base value for logarithmic axis| **Property:** *logBase*

var axes= {logBase: 10};
<EJ.Chart
axes={axes}>
</EJ.Chart>| **Property:** *logBase*

<ChartComponent id='charts'>
<AxisDirective logBase: 10>
</AxisDirective>
</ChartComponent>|

|Major grid line| **Property:** *majorGridLines.visible*

var axes= {majorGridLines: { visible: true}};
<EJ.Chart
axes={axes}>
</EJ.Chart>| Not Applicable |

|Width of MajorGridLines| **Property:** *majorGridLines.width*

var axes= {majorGridLines: { width: 2}};
<EJ.Chart
axes={axes}>
</EJ.Chart>| **Property:** *majorGridLines.width*

<ChartComponent id='charts'>
<AxisDirective majorGridLines: { width: 2}>
</AxisDirective>
</ChartComponent>|

|Color of MajorGridLines| **Property:** *majorGridLines.color*

var axes= {majorGridLines: { color: 'black' }};
<EJ.Chart
axes={axes}>
</EJ.Chart>| **Property:** *majorGridLines.color*

<ChartComponent id='charts'>
<AxisDirective majorGridLines: { color: 'black' }>
</AxisDirective>
</ChartComponent>|

|DashArray of MajorGridLines| **Property:** *majorGridLines.dashArray*

var axes= {majorGridLines: { dashArray: '10,5' }};
<EJ.Chart
axes={axes}>
</EJ.Chart>| **Property:** *majorGridLines.dashArray*

<ChartComponent id='charts'>
<AxisDirective majorGridLines: { dashArray: '10,5' }>
</AxisDirective>
</ChartComponent>|

|Opacity of major grid line| **Property:** *majorGridLines.opacity*

var axes= {majorGridLines: { opacity: true }};
<EJ.Chart
axes={axes}>
</EJ.Chart>| Not Applicable |

|Major Tick line| **Property:** *majorTickLines.visible*

var axes= {majorTickLines: { visible: true }};
<EJ.Chart
axes={axes}>
</EJ.Chart>| Not Applicable |

|Width of MajorTickLines| **Property:** *majorTickLines.width*

var axes= {majorTickLines: { width: 2 }};
<EJ.Chart
axes={axes}>
</EJ.Chart>| **Property:** *majorTickLines.width*

<ChartComponent id='charts'>
<AxisDirective majorTickLines: { width: 2 }>
</AxisDirective>
</ChartComponent>|

| Height of MajorTickLines| **Property:** *majorTickLines.size*

var axes= {majorTickLines: { size: 2 }};
<EJ.Chart
axes={axes}>
</EJ.Chart>| **Property:** *majorTickLines.height*

<ChartComponent id='charts'>
<AxisDirective majorTickLines: { height: 2}>
</AxisDirective>
</ChartComponent>|

| Color of MajorTickLines| **Property:** *majorTickLines.color*

var axes= {majorTickLines: { color: 'black' }};
<EJ.Chart
axes={axes}>
</EJ.Chart>| **Property:** *majorTickLines.color*

<ChartComponent id='charts'>
<AxisDirective majorTickLines: { color: 'black' }>
</AxisDirective>
</ChartComponent>|

| Opacity of major Tick line| **Property:** *majorTickLines.opacity*

var axes= {majorTickLines: { opacity: true}};
<EJ.Chart
axes={axes}>
</EJ.Chart>| Not Applicable|

| maximum labels of axes| **Property:** *maximumLabels*

var axes= { maximumLabels: 5};
<EJ.Chart
axes={axes}>
</EJ.Chart>| **Property:** *maximumLabels*

<ChartComponent id='charts'>
<AxisDirective maximumLabels: 4>
</AxisDirective>
</ChartComponent>|

| maximum labels width of axes to trim| **Property:** *maximumLabelWidth*

var axes= { maximumLabelWidth: 40};
<EJ.Chart
axes={axes}>
</EJ.Chart>| **Property:** *maximumLabelWidth*

<ChartComponent id='charts'>
<AxisDirective maximumLabelWidth: 40>
</AxisDirective>
</ChartComponent>|

| minor grid line| **Property:** *minorGridLines.visible*

var axes= { minorGridLines: { visible: true}};
<EJ.Chart
axes={axes}>
</EJ.Chart>| Not Applicable|

| Width of minorGridLines| **Property:** *minorGridLines.width*

var axes= { minorGridLines: { width: 2}};
<EJ.Chart
axes={axes}>
</EJ.Chart>| **Property:** *minorGridLines.width*

<ChartComponent id='charts'>
<AxisDirective minorGridLines: { width: 2}>
</AxisDirective>
</ChartComponent>|

| Color of minorGridLines| **Property:** *minorGridLines.color*

var axes= { minorGridLines: { color: 'black'}};
<EJ.Chart
axes={axes}>
</EJ.Chart>| **Property:** *minorGridLines.color*

<ChartComponent id='charts'>
<AxisDirective minorGridLines: { color: 'black'}>
</AxisDirective>
</ChartComponent>|

| DashArray of minorGridLines| **Property:** *minorGridLines.dashArray*

var axes= { minorGridLines: { dashArray: '10,5'}};
<EJ.Chart
axes={axes}>
</EJ.Chart>| **Property:** *minorGridLines.dashArray*

<ChartComponent id='charts'>
<AxisDirective minorGridLines: { dashArray: '10,5' }>
</AxisDirective>
</ChartComponent>|

| Opacity of minor grid line| **Property:** *minorGridLines.opacity*

var axes= { minorGridLines: { opacity: true}};
<EJ.Chart
axes={axes}>
</EJ.Chart>| Not Applicable|

| minor Tick line| **Property:** *minorTickLines.visible*

var axes= { minorTickLines: { visible: true}};
<EJ.Chart
axes={axes}>
</EJ.Chart>| Not Applicable|

| Width of minorTickLines| **Property:** *minorTickLines.width*

var axes= { minorTickLines: { width: 2 }};
<EJ.Chart
axes={axes}>
</EJ.Chart>| **Property:** *minorTickLines.width*

<ChartComponent id='charts'>
<AxisDirective minorTickLines: { width: 2}>
</AxisDirective>
</ChartComponent>|

|Height of minorTickLines| **Property:** *minorTickLines.size*

var axes= { minorTickLines: { size: 2}};
<EJ.Chart
axes={axes}>
</EJ.Chart>|**Property:** *minorTickLines.height*

<ChartComponent id='charts'>
<AxisDirective minorTickLines: { height: 2}>
</AxisDirective>
</ChartComponent>|

|Color of minorTickLines| **Property:** *minorTickLines.color*

var axes= { minorTickLines: { color: 'black' }};
<EJ.Chart
axes={axes}>
</EJ.Chart>|**Property:** *minorTickLines.color*

<ChartComponent id='charts'>
<AxisDirective minorTickLines: { color: 'black' }>
</AxisDirective>
</ChartComponent>|

|Opacity of minor Tick line| **Property:** *minorTickLines.opacity*

var axes= { minorTickLines: { opacity: true }};
<EJ.Chart
axes={axes}>
</EJ.Chart>|Not Applicable|

|Minor ticks per interval of axes| **Property:** *minorTicksPerInterval*

var axes= { minorTicksPerInterval: 4};
<EJ.Chart
axes={axes}>
</EJ.Chart>|**Property:** *minorTicksPerInterval*

<ChartComponent id='charts'>
<AxisDirective minorTicksPerInterval: 4>
</AxisDirective>
</ChartComponent>|

|name of the axes| **Property:** *name*

var axes= { name: 'axes'};
<EJ.Chart
axes={axes}>
</EJ.Chart>|**Property:** *name*

<ChartComponent id='charts'>
<AxisDirective name: 'primaryYAxis'>
</AxisDirective>
</ChartComponent>|

|Orientation of axes| **Property:** *orientation*

var axes= { orientation: 'Vertical'};
<EJ.Chart
axes={axes}>
</EJ.Chart>|Not Applicable|

|Plot offset for axes| **Property:** *plotOffset*

var axes= { plotOffset: 0};
<EJ.Chart
axes={axes}>
</EJ.Chart>|**Property:** *plotOffset*

<ChartComponent id='charts'>
<AxisDirective plotOffset: 0>
</AxisDirective>
</ChartComponent>|

|minimum for axes| **Property:** *range.minimum*

var axes= { range: { minimum: 10 }};
<EJ.Chart
axes={axes}>
</EJ.Chart>|**Property:** *minimum*

<ChartComponent id='charts'>
<AxisDirective minimum: 23>
</AxisDirective>
</ChartComponent>|

|maximum for axes| **Property:** *range.maximum*

var axes= { range: { maximum: 10 }};
<EJ.Chart
axes={axes}>
</EJ.Chart>|**Property:** *maximum*

<ChartComponent id='charts'>
<AxisDirective maximum: 23>
</AxisDirective>
</ChartComponent>|

|interval for axes| **Property:** *range.interval*

var axes= { range: { interval: 1 }};
<EJ.Chart
axes={axes}>
</EJ.Chart>|**Property:** *interval*

<ChartComponent id='charts'>
<AxisDirective interval: 2>
</AxisDirective>
</ChartComponent>|

|RangePadding for axes| **Property:** *rangePadding*

var axes= { rangePadding: 'None'};
<EJ.Chart
axes={axes}>
</EJ.Chart>|**Property:** *rangePadding*

<ChartComponent id='charts'>
<AxisDirective rangePadding: 'None'>
</AxisDirective>
</ChartComponent>|

| Rounding Places in axes | **Property:** *roundingPlaces*

var axes= { roundingPlaces: 3};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *labelFormat*

<ChartComponent id='charts'>
<AxisDirective labelFormat: 'n3'>
</AxisDirective>
</ChartComponent> |

| ScrollBar settings of axes | **Property:** *scrollbarSettings*

var axes= { scrollbarSettings : { }};
<EJ.Chart
axes={axes}>
</EJ.Chart> | Not Applicable |

| TickPosition in axes | **Property:** *tickLinesPosition*

var axes= { tickLinesPosition: 'Inside'};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *tickPosition*

<ChartComponent id='charts'>
<AxisDirective tickPosition: 'Inside'>
</AxisDirective>
</ChartComponent> |

| valueType of axes | **Property:** *valueType*

var axes= { valueType: 'DateTime'};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *valueType*

<ChartComponent id='charts'>
<AxisDirective valueType: 'DateTime'>
</AxisDirective>
</ChartComponent> |

| visible of axes | **Property:** *visible*

var axes= { visible: true};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *visible*

<ChartComponent id='charts'>
<AxisDirective visible: true>
</AxisDirective>
</ChartComponent> |

| zoomFactor of axes | **Property:** *zoomFactor*

var axes= { zoomFactor: 0.3};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *zoomFactor*

<ChartComponent id='charts'>
<AxisDirective zoomFactor: 0.3>
</AxisDirective>
</ChartComponent> |

| zoomPosition of axes | **Property:** *zoomPosition*

var axes= { zoomPosition: 0.3};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *zoomPosition*

<ChartComponent id='charts'>
<AxisDirective zoomPosition: 0.3>
</AxisDirective>
</ChartComponent> |

| labelBorder of axes | **Property:** *labelBorder*

var axes= { labelBorder: { color: 'red', width: 2}};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *border*

<ChartComponent id='charts'>
<AxisDirective border: { color: 'red', width: 3 }>
</AxisDirective>
</ChartComponent> |

| title of axes | **Property:** *title.text*

var axes= { title: { text: 'Chart title' }};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *title*

<ChartComponent id='charts'>
<AxisDirective title: 'Chart title'>
</AxisDirective>
</ChartComponent> |

| StripLine of axes | **Property:** *stripLine*

var axes= { stripLine: []};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *stripLines*

<ChartComponent id='charts'>
<AxisDirective stripLines: []>
</AxisDirective>
</ChartComponent> |

| Multilevel labels of axes | **Property:** *multiLevelLabels*

var axes= { multiLevelLabels: []};
<EJ.Chart
axes={axes}>
</EJ.Chart> | **Property:** *multiLevelLabels*

```
<br/><br/><ChartComponent id='charts'><br/><AxisDirective multiLevelLabels:
[]><br/></AxisDirective><br/></ChartComponent>|
```

```
| skeleton for an axes | Not Applicable | Property: axes.skeleton <br/><br/><ChartComponent
id='charts'><br/><AxisDirective skeleton: 'yMd'
><br/></AxisDirective><br/></ChartComponent>|
```

```
| skeleton type for an axes | Not Applicable | Property: axes.skeletonType <br/><br/><ChartComponent
id='charts'><br/><AxisDirective skeletonType: 'DateTime'
><br/></AxisDirective><br/></ChartComponent>|
```

Rows

```
<!-- markdownlint-disable MD033 -->
```

```
| Behavior | API in Essential JS 1 | API in Essential JS 2 |
```

```
| --- | --- | --- |
```

```
| rows in chart | Property: rowDefinitions <br/><br/>var rowDefinitions=[{
}];<br/><EJ.Chart<br/>rowDefinitions={rowDefinitions}> <br/></EJ.Chart> | Property: rowDirective
<br/><br/><ChartComponent
id='charts'><br/><rowDirective><br/></rowDirective><br/></ChartComponent>|
```

```
| Unit | Property: unit <br/><br/>var rowDefinitions=[{ unit : 'percentage'
}];<br/><EJ.Chart<br/>rowDefinitions={rowDefinitions}> <br/></EJ.Chart> | Not Applicable|
```

```
| height of rows in chart | Property: rowHeight <br/><br/>var rowDefinitions=[{rowHeight :
50}];<br/><EJ.Chart<br/>rowDefinitions={rowDefinitions}> <br/></EJ.Chart> | Property: height
<br/><br/><ChartComponent id='charts'><br/><RowDirective height='50%'
><br/></RowDirective><br/></ChartComponent>|
```

```
| Line customization | Property: lineColor, lineWidth <br/><br/>var rowDefinitions=[{lineColor : 'Gray',
linewidth : 0}];<br/><EJ.Chart<br/>rowDefinitions={rowDefinitions}> <br/></EJ.Chart> | Property:
border <br/><br/><ChartComponent id='charts'><br/><RowDirective border={width: 2,
color:'brown'}><br/></RowDirective><br/></ChartComponent>|
```

Series

```
<!-- markdownlint-disable MD033 -->
```

```
| Behavior | API in Essential JS 1 | API in Essential JS 2 |
```

```
| --- | --- | --- |
```

```
| bearFillColor | Property: bearFillColor <br/><br/>var series =[{ bearFillColor:
'red'}];<br/><EJ.Chart<br/>series={series}> <br/></EJ.Chart> | Property: bearFillColor
<br/><br/><ChartComponent id='charts'><br/><SeriesDirective
bearFillColor='red'><br/></SeriesDirective><br/></ChartComponent>|
```

```
| Border | Property: border <br/><br/>var series =[{ border: {color: 'red', width:
2}}];<br/><EJ.Chart<br/>series={series}> <br/></EJ.Chart> | Property: border
<br/><br/><ChartComponent id='charts'><br/><SeriesDirective border={ color: 'red', width:
2}><br/></SeriesDirective><br/></ChartComponent>|
```


|BoxPlotMode| **Property:** *boxPlotMode*

var series =[{ boxPlotMode:
'inclusive'}]
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:** *boxPlotMode*

<ChartComponent id='charts'>
<SeriesDirective
boxPlotMode='Inclusive'>
</SeriesDirective>
</ChartComponent>|

|Minimum radius of Bubble series| **Property:** *bubbleOptions.minRadius*

var series =[{
bubbleOptions: { minRadius: 2}}]
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:**
minRadius

<ChartComponent id='charts'>
<SeriesDirective
minRadius='2'>
</SeriesDirective>
</ChartComponent>|

|Maximum radius of Bubble series| **Property:** *bubbleOptions.maxRadius*

var series =[{
bubbleOptions: { maxRadius: 2}}]
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:**
maxRadius

<ChartComponent id='charts'>
<SeriesDirective
minRadius='2'>
</SeriesDirective>
</ChartComponent>|

|bullFillColor| **Property:** *bullFillColor*

var series =[{ bullFillColor:
'red'}]
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:** *bullFillColor*

<ChartComponent id='charts'>
<SeriesDirective
bullFillColor='red'>
</SeriesDirective>
</ChartComponent>|

|Cardinal spline tension for spline series| **Property:** *cardinalSplineTension*

var series =[{
cardinalSplineTension: 0.5}]
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:**
cardinalSplineTension

<ChartComponent id='charts'>
<SeriesDirective
cardinalSplineTension='0.5'>
</SeriesDirective>
</ChartComponent>|

|Column Width for rectangle series| **Property:** *columnWidth*

var series =[{ columnWidth :
0.5}]
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:** *cardinalSplineTension*

<ChartComponent id='charts'>
<SeriesDirective
columnWidth='0.5'>
</SeriesDirective>
</ChartComponent>|

|Column spacing for rectangle series| **Property:** *columnSpacing*

var series =[{ columnSpacing:
0.5}]
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:** *columnSpacing*

<ChartComponent id='charts'>
<SeriesDirective
columnSpacing='0.5'>
</SeriesDirective>
</ChartComponent>|

|Topleft radius for rectangle series| **Property:** *cornerRadius.topLeft*

var series =[{ topLeft:
0}]
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:** *cornerRadius.topLeft*

<ChartComponent id='charts'>
<SeriesDirective
topLeft='0.5'>
</SeriesDirective>
</ChartComponent>|

|topRight radius for rectangle series| **Property:** *cornerRadius.topRight*

var series =[{ topRight:
0}]
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:** *cornerRadius.topRight*

<ChartComponent id='charts'>
<SeriesDirective topRight:
0>
</SeriesDirective>
</ChartComponent>|

|bottomRight radius for rectangle series| **Property:** *cornerRadius.bottomRight*

var series =[{
bottomRight: 0}]
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:**
cornerRadius.bottomRight

<ChartComponent id='charts'>
<SeriesDirective
bottomRight: 0>
</SeriesDirective>
</ChartComponent>|

|bottomLeft radius for rectangle series| **Property:** *cornerRadius.bottomLeft*

var series =[{
bottomLeft: 0}]
<EJ.Chart
series={series}>
</EJ.Chart>| **Property:**
cornerRadius.bottomLeft

<ChartComponent id='charts'>
<SeriesDirective
bottomLeft: 0>
</SeriesDirective>
</ChartComponent>|

|DashArray property| **Property:** *dashArray*

var series =[{ dashArray: '10,
5'}]
<EJ.Chart
series={series}>
</EJ.Chart>| **Property:** *dashArray*

<ChartComponent id='charts'>
<SeriesDirective dashArray='10,
5'>
</SeriesDirective>
</ChartComponent>|

|DataSource for series| **Property:** *dataSource*

var series =[{ dataSource: {}
}]
<EJ.Chart
series={series}>
</EJ.Chart>| **Property:** *dataSource*

<ChartComponent id='charts'>
<SeriesDirective dataSource:
[]>
</SeriesDirective>
</ChartComponent>|

|Draw type for Polar series| **Property:** *drawType*

var series =[{ drawType:
'Line'}]
<EJ.Chart
series={series}>
</EJ.Chart>| **Property:** *drawType*

<ChartComponent id='charts'>
<SeriesDirective
drawType='Line'>
</SeriesDirective>
</ChartComponent>|

|EmptyPointSettings for series| **Property:** *emptyPointSettings.visible*

var series
=[{emptyPointSettings: { visible: false }}]
<EJ.Chart
series={series}>
</EJ.Chart>| Not
Applicable|

|Empty Point Display mode| **Property:** *emptyPointSettings.displayMode*

var series
=[{emptyPointSettings: { displayMode: 'gap' }}]
<EJ.Chart
series={series}>

</EJ.Chart>| **Property:** *emptyPointSettings.displayMode*

<ChartComponent
id='charts'>
<SeriesDirective emptyPointSettings={displayMode:
'gap'}>
</SeriesDirective>
</ChartComponent>|

|Empty Point color| **Property:** *emptyPointSettings.color*

var series =[{ emptyPointSettings: {
color: 'red'}}]
<EJ.Chart
series={series}>
</EJ.Chart>| **Property:**
emptyPointSettings.color

<ChartComponent id='charts'>
<SeriesDirective
emptyPointSettings={color: 'red'}>
</SeriesDirective>
</ChartComponent>|

|Empty Point Border| **Property:** *emptyPointSettings.border*

var series =[{emptyPointSettings:
{ border: {}}}]
<EJ.Chart
series={series}>
</EJ.Chart>| **Property:**
emptyPointSettings.border

<ChartComponent id='charts'>
<SeriesDirective
emptyPointSettings={border: {}}>
</SeriesDirective>
</ChartComponent>|

|Enable animation for series| **Property:** *enableAnimation*

var series =[{enableAnimation :
true}]
<EJ.Chart
series={series}>
</EJ.Chart>| **Property:** *animation.enable*

<ChartComponent id='charts'>
<SeriesDirective animation={
enable={false}}>
</SeriesDirective>
</ChartComponent>|

|Animation duration for series| **Property:** *animationDuration*

var series
=[{animationDuration: 1000}]
<EJ.Chart
series={series}>
</EJ.Chart>| **Property:**
animation.duration

<ChartComponent id='charts'>
<SeriesDirective animation={
duration={1000}}>
</SeriesDirective>
</ChartComponent>|

| Animation delay for series | Not Applicable | **Property:** *animation.delay*

<ChartComponent id='charts'>
<SeriesDirective animation={ delay={100}>
</SeriesDirective>
</ChartComponent> |

| Drag settings for series | **Property:** *dragSettings*

var series =[{dragSettings : { mode: 'X' }}]
<EJ.Chart
series={series}>
</EJ.Chart> | Not Applicable |

| Errorbar settings for series | **Property:** *errorBarSettings*

var series =[{errorBarSettings: { }}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *errorBarSettings*

<ChartComponent id='charts'>
<SeriesDirective errorBarSettings={>
</SeriesDirective>
</ChartComponent> |

| Closed series | **Property:** *isClosed*

var series =[{isClosed: true}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *isClosed*

<ChartComponent id='charts'>
<SeriesDirective isClosed={true}>
</SeriesDirective>
</ChartComponent> |

| Stacking Property for series | **Property:** *isStacking*

var series =[{isStacking : true}]
<EJ.Chart
series={series}>
</EJ.Chart> | Not Applicable |

| Line cap for series | **Property:** *lineCap*

var series =[{lineCap : 'butt'}]
<EJ.Chart
series={series}>
</EJ.Chart> | Not Applicable |

| Line join for series | **Property:** *lineJoin*

var series =[{lineJoin : 'round'}]
<EJ.Chart
series={series}>
</EJ.Chart> | Not Applicable |

| Opacity for series | **Property:** *opacity*

var series =[{opacity: 0.7}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *opacity*

<ChartComponent id='charts'>
<SeriesDirective opacity={0.7}>
</SeriesDirective>
</ChartComponent> |

| Outlier settings of series | **Property:** *outLierSettings*

var series =[{outLierSettings: { }}]
<EJ.Chart
series={series}>
</EJ.Chart> | Not Applicable |

| Palette | **Property:** *palette*

var series =[{palette: "ColorFieldName"}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *pointColorMapping*

<ChartComponent id='charts'>
<SeriesDirective pointColorMapping='color'>
</SeriesDirective>
</ChartComponent> |

| Positive fill for waterfall series | **Property:** *positiveFill*

var series =[{ positiveFill: "red"}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *positiveFill*

<ChartComponent id='charts'>
<SeriesDirective positiveFill: "red">
</SeriesDirective>
</ChartComponent> |

| Show average value in box and whisker series | **Property:** *showMedian*

var series =[{ showMedian: true}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *showMean*

<ChartComponent id='charts'>
<SeriesDirective showMean={false}>
</SeriesDirective>
</ChartComponent> |

| To group the series of stacking collection | **Property:** *stackingGroup*

var series =[{ stackingGroup: 'group'}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:**

```
stackingGroup <br/><br/><ChartComponent id='charts'><br/><SeriesDirective
stackingGroup='group'><br/></SeriesDirective><br/></ChartComponent>|
```

| Specifies the type of the series to render in chart. | **Property:** *type*

var series = [{ type: 'Line'}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *type*

<ChartComponent id='charts'>
<SeriesDirective type: 'Line'>
</SeriesDirective>
</ChartComponent>|

| Defines the visibility of the series | **Property:** *visibility*

var series = [{ visibility: true}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *visible*

<ChartComponent id='charts'>
<SeriesDirective visible={true}>
</SeriesDirective>
</ChartComponent>|

| Specifies the different types of spline curve | **Property:** *splineType*

var series = [{ splineType : 'Natural'}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *splineType*

<ChartComponent id='charts'>
<SeriesDirective splineType='Natural'>
</SeriesDirective>
</ChartComponent>|

| Specifies the name of the x-axis that has to be associated with this series. Add an axis instance with this name to axes collection. | **Property:** *xAxisName*

var series = [{xAxisName : 'secondaryXAxis'}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *xAxisName*

<ChartComponent id='charts'>
<SeriesDirective xAxisName='secondaryXAxis'>
</SeriesDirective>
</ChartComponent>|

| Name of the property in the datasource that contains x value for the series. | **Property:** *xName*

var series = [{xName : 'x'}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *xName*

<ChartComponent id='charts'>
<SeriesDirective xName='x'>
</SeriesDirective>
</ChartComponent>|

| Specifies the name of the y-axis that has to be associated with this series. Add an axis instance with this name to axes collection. | **Property:** *yAxisName*

var series = [{yAxisName : 'secondaryYAxis'}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *yAxisName*

<ChartComponent id='charts'>
<SeriesDirective yAxisName='secondaryYAxis'>
</SeriesDirective>
</ChartComponent>|

| Name of the property in the datasource that contains y value for the series. | **Property:** *yName*

var series = [{yName : 'y'}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *yName*

<ChartComponent id='charts'>
<SeriesDirective yName='y'>
</SeriesDirective>
</ChartComponent>|

| Name of the property in the datasource that contains high value for the series. | **Property:** *high*

var series = [{high : 'y'}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *high*

<ChartComponent id='charts'>
<SeriesDirective high='y'>
</SeriesDirective>
</ChartComponent>|

| Name of the property in the datasource that contains low value for the series. | **Property:** *low*

var series = [{low : 'y'}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *low*

<ChartComponent id='charts'>
<SeriesDirective low='y'>
</SeriesDirective>
</ChartComponent>|

| Name of the property in the datasource that contains close value for the series. | **Property:** *close*

var series = [{close : 'y'}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:**
close

<ChartComponent id='charts'>
<SeriesDirective
 close='y'>
</SeriesDirective>
</ChartComponent> |

| Name of the property in the datasource that contains open value for the series | **Property:** *open*

var series = [{open : 'y'}]
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:**
open

<ChartComponent id='charts'>
<SeriesDirective
 open='y'>
</SeriesDirective>
</ChartComponent> |

| Option to add trendlines to chart | **Property:** *trendLines*

var series = [{trendLines :
 [{}]}
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *trendLines*

<ChartComponent id='charts'>
<SeriesDirective
 trendLines=[{}]>
</SeriesDirective>
</ChartComponent> |

| Options for customizing the appearance of the series or data point while highlighting | **Property:**
highlightSettings

var series = [{ highlightSettings : {} }]
<EJ.Chart
series={series}>

</EJ.Chart> | Not applicable. |

| Options for customizing the appearance of the series/data point on selection | **Property:**
selectionSettings

var series = [{ selectionSettings : {} }]
<EJ.Chart
series={series}>

</EJ.Chart> | Not applicable. |

marker

<!-- markdownlint-disable MD033 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| visibility of marker | **Property:** *marker.visible*

var series = [{marker: { visible: true
 }}
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *marker.visible*

<ChartComponent id='charts'>
<SeriesDirective marker={ visible: false
 }>
</SeriesDirective>
</ChartComponent> |

| visibility of marker | **Property:** *marker.visible*

var series = [{marker: { visible: true
 }}
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *marker.visible*

<ChartComponent id='charts'>
<SeriesDirective marker={ visible: false
 }>
</SeriesDirective>
</ChartComponent> |

| Fill for marker | **Property:** *marker.fill*

var series = [{marker: { fill : 'red'
 }}
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *marker.fill*

<ChartComponent id='charts'>
<SeriesDirective marker={ fill : 'red'
 }>
</SeriesDirective>
</ChartComponent> |

| Opacity for marker | **Property:** *marker.opacity*

var series = [{marker: { opacity : 0.5
 }}
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *marker.opacity*

<ChartComponent id='charts'>
<SeriesDirective marker={ opacity : 0.5
 }>
</SeriesDirective>
</ChartComponent> |

| Shape of marker | **Property:** *marker.shape*

var series = [{marker: { shape :
 'Circle'}}
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *marker.shape*

```
<br/><br/><ChartComponent id='charts'><br/><SeriesDirective marker={ shape : 'Triangle'
}><br/></SeriesDirective><br/></ChartComponent>|
```

|ImageUrl of marker| **Property:** *imageUrl*

var series = [{marker: { imageUrl :

```
""}]<br/><EJ.Chart<br/>series={series}> <br/></EJ.Chart>|Property: imageUrl
<br/><br/><ChartComponent id='charts'><br/><SeriesDirective marker={ imageUrl : "
}><br/></SeriesDirective><br/></ChartComponent>|
```

|Border of marker| **Property:** *border*

var series = [{marker: { border : { width: 2, color: 'red'
}}}]
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:** *border*

<ChartComponent id='charts'>
<SeriesDirective marker={ border : { width: 2,
color: 'red'}}>
</SeriesDirective>
</ChartComponent>|

|Height of marker| **Property:** *size.height*

var series = [{marker:{ size: { height: 30}}
}]
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:** *height*

<ChartComponent id='charts'>
<SeriesDirective marker={ height:
25}>
</SeriesDirective>
</ChartComponent>|

|Width of marker| **Property:** *size.width*

var series = [{marker:{ size: { width: 30}}
}]
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:** *width*

<ChartComponent id='charts'>
<SeriesDirective marker={ width:
30}>
</SeriesDirective>
</ChartComponent>|

|DataLabelSettings of marker| **Property:** *marker.dataLabel*

var series = [{marker:{ dataLabel: {
}}}]
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:** *marker.dataLabel*

<ChartComponent id='charts'>
<SeriesDirective marker={ dataLabel: {
}}>
</SeriesDirective>
</ChartComponent>|

|Visibility of dataLabel| **Property:** *dataLabel.visible*

var series = [{marker:{ dataLabel: { visible:
true }} }]
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:** *dataLabel.visible*

<ChartComponent id='charts'>
<SeriesDirective marker={ dataLabel: { visible:
true }}>
</SeriesDirective>
</ChartComponent>|

|Text mapping name of dataLabel| **Property:** *dataLabel.textMappingName*

var series
= [{marker:{ dataLabel: { textMappingName: " }} }]
<EJ.Chart
series={series}>

</EJ.Chart>|**Property:** *dataLabel.name*

<ChartComponent
id='charts'>
<SeriesDirective marker={ dataLabel: { name: "
}}>
</SeriesDirective>
</ChartComponent>|

|Fill color of data label| **Property:** *dataLabel.fill*

var series = [{marker:{ dataLabel: { fill: 'pink' }}
}]
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:** *dataLabel.fill*

<ChartComponent id='charts'>
<SeriesDirective marker={ dataLabel: { fill: 'pink'
}}>
</SeriesDirective>
</ChartComponent>|

|Opacity of data label| **Property:** *dataLabel.opacity*

var series = [{marker:{ dataLabel: {
opacity: 0.6 }} }]
<EJ.Chart
series={series}>
</EJ.Chart>|**Property:** *dataLabel.opacity*

<ChartComponent id='charts'>
<SeriesDirective marker={ dataLabel: { opacity:
0.6 }}>
</SeriesDirective>
</ChartComponent>|

|Text position of data label| **Property:** *dataLabel.textPosition*

var series = [{marker:{
dataLabel: {textPosition: 'middle' }} }]
<EJ.Chart
series={series}>

</EJ.Chart>| **Property:** *dataLabel.position*

<ChartComponent
id='charts'>
<SeriesDirective marker={ dataLabel: { position: 'Top'
}}>
</SeriesDirective>
</ChartComponent>|

|Alignment of data label| **Property:** *dataLabel.verticalAlignment*

var series = [{marker:{
dataLabel: {verticalAlignment: 'near' }} }]
<EJ.Chart
series={series}>

</EJ.Chart>| **Property:** *dataLabel.alignment*

<ChartComponent
id='charts'>
<SeriesDirective marker={ dataLabel: { alignment: 'Near'
}}>
</SeriesDirective>
</ChartComponent>|

|Border of data label| **Property:** *dataLabel.border*

var series = [{marker:{ dataLabel: {border:
{}} }]
<EJ.Chart
series={series}>
</EJ.Chart>| **Property:** *dataLabel.border*

<ChartComponent id='charts'>
<SeriesDirective marker={ dataLabel: { border: {}
}}>
</SeriesDirective>
</ChartComponent>|

|Offset for data label| **Property:** *dataLabel.offset*

var series = [{marker:{ dataLabel: {offset: {
x: 5, y: 6 }} }]
<EJ.Chart
series={series}>
</EJ.Chart>| Not Applicable|

|Margin of data label| **Property:** *dataLabel.margin*

var series = [{marker:{ dataLabel: {margin:
{}} }]
<EJ.Chart
series={series}>
</EJ.Chart>| **Property:** *dataLabel.margin*

<ChartComponent id='charts'>
<SeriesDirective marker={ dataLabel: { margin: {}
}}>
</SeriesDirective>
</ChartComponent>|

|Font of data label| **Property:** *dataLabel.font*

var series = [{marker:{ dataLabel: {font: {}}
}}
<EJ.Chart
series={series}>
</EJ.Chart>| **Property:** *dataLabel.font*

<ChartComponent id='charts'>
<SeriesDirective marker={ dataLabel: { font: {}
}}>
</SeriesDirective>
</ChartComponent>|

|HTML template in dataLabel| **Property:** *dataLabel.template*

var series = [{marker:{
dataLabel: {template: '<div>Chart</div>' }} }]
<EJ.Chart
series={series}>

</EJ.Chart>| **Property:** *dataLabel.template*

<ChartComponent
id='charts'>
<SeriesDirective marker={ dataLabel: { template: '<div>Chart</div>'
}}>
</SeriesDirective>
</ChartComponent>|

|Rounded corner radius X| Not Applicable| **Property:** *dataLabel.rx*

<ChartComponent
id='charts'>
<SeriesDirective marker={ dataLabel: { rx: 10
}}>
</SeriesDirective>
</ChartComponent>|

|Rounded corner radius Y| Not Applicable| **Property:** *dataLabel.ry*

<ChartComponent
id='charts'>
<SeriesDirective marker={ dataLabel: { ry: 10
}}>
</SeriesDirective>
</ChartComponent>|

|Maximum Label width for data label| **Property:** *dataLabel.maximumLabelWidth*

var series
= [{marker:{ dataLabel: {maximumLabelWidth: 20 }} }]
<EJ.Chart
series={series}>

</EJ.Chart>| Not Applicable|

| Enable wrapping of text for data label | **Property:** *dataLabel.enableWrap*

var series
 =[{marker:{ dataLabel: {enableWrap: true} }}
<EJ.Chart
series={series}>

</EJ.Chart> | Not Applicable |

| To show contrast color for data label | **Property:** *dataLabel.showContrastColor*

var series
 =[{marker:{ dataLabel: {showContrastColor: true} }}
<EJ.Chart
series={series}>

</EJ.Chart> | Not Applicable |

| To show edge label for data label | **Property:** *dataLabel.showEdgeLabels*

var series
 =[{marker:{ dataLabel: {showEdgeLabels: true} }}
<EJ.Chart
series={series}>

</EJ.Chart> | Not Applicable |

TrendLines

<!-- markdownlint-disable MD033 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| Trendlines settings | **Property:** *series.trendLines*

var series =[{trendlines: [{ visibility: "visible",
 type: "linear" }]}
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *series.trendLines*

<ChartComponent
 id='charts'>
<TrendlineDirective>
</TrendlineDirective>
</ChartComponent> |

| Visibility of trendline | **Property:** *trendLines.visibility*

var series =[{trendlines: [{ visibility:
 "visible"}]}
<EJ.Chart
series={series}>
</EJ.Chart> | Not applicable |

| Type of trendLine | **Property:** *trendLines.type*

var series =[{trendlines: [{type: "linear" }]
 }
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *trendLines.type*

<ChartComponent id='charts'>
<TrendlineDirective
 type='Linear'>
</TrendlineDirective>
</ChartComponent> |

| Name of trendLine | **Property:** *trendLines.name*

var series =[{trendlines: [{ name: 'trendLine'
 }]}
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *trendLines.name*

<ChartComponent id='charts'>
<TrendlineDirective
 name='trendLine'>
</TrendlineDirective>
</ChartComponent> |

| Period of trendLine | **Property:** *trendLines.period*

var series =[{trendlines: [{ period: 45}]
 }
<EJ.Chart
series={series}>
</EJ.Chart> | **Property:** *trendLines.period*

<ChartComponent id='charts'>
<TrendlineDirective
 name='trendLine'>
</TrendlineDirective>
</ChartComponent> |

| Polynomial order for Polynomial type trendLines | **Property:** *trendLines.polynomialOrder*

var
 series =[{trendlines: [{ polynomialOrder: 3}]}
<EJ.Chart
series={series}>

</EJ.Chart> | **Property:** *trendLines.polynomialOrder*

<ChartComponent
 id='charts'>
<TrendlineDirective
 polynomialOrder={3}>
</TrendlineDirective>
</ChartComponent> |

| Backward forecast for trendLines | **Property:** *trendLines.backwardforecast*

var series
 =[{trendlines: [{ backwardforecast: 3 }]}
<EJ.Chart
series={series}>

</EJ.Chart> | **Property:** *trendLines.backwardforecast*

<ChartComponent


```

id='charts'><br/><TrendlineDirective
backwardforecast={3}><br/></TrendlineDirective><br/></ChartComponent>|
| Forward forecast for trendLines| Property: trendLines.forwardforecast <br/><br/>var series
=[{trendlines: [{ forwardforecast: 3 }]}]><br/><EJ.Chart><br/>series={series}>
<br/></EJ.Chart>| Property: trendLines.forwardforecast <br/><br/><ChartComponent
id='charts'><br/><TrendlineDirective
forwardforecast={3}><br/></TrendlineDirective><br/></ChartComponent>|
| Fill for trendLines| Property: trendLines.fill <br/><br/>var series =[{trendlines: [{ fill: '#EEFFCC'}]}]
><br/><EJ.Chart><br/>series={series}> <br/></EJ.Chart>| Property: trendLines.fill
<br/><br/><ChartComponent id='charts'><br/><TrendlineDirective
fill='EEFFCC'><br/></TrendlineDirective><br/></ChartComponent>|
| Width for trendLines| Property: trendLines.width <br/><br/>var series =[{trendlines: [{ width: 2 }]}]
><br/><EJ.Chart><br/>series={series}> <br/></EJ.Chart>| Property: trendLines.width
<br/><br/><ChartComponent id='charts'><br/><TrendlineDirective
width={2}><br/></TrendlineDirective><br/></ChartComponent>|
| Intercept value for trendLines| Property: trendLines.intercept <br/><br/>var series =[{trendlines: [{
intercept: 2 }]}]><br/><EJ.Chart><br/>series={series}> <br/></EJ.Chart>| Property: trendLines.intercept
<br/><br/><ChartComponent id='charts'><br/><TrendlineDirective
intercept={2}><br/></TrendlineDirective><br/></ChartComponent>|
| Legend shape for trendLines| Not Applicable| Property: trendLines.legendShape
<br/><br/><ChartComponent id='charts'><br/><TrendlineDirective
legendShape='Rectangle'><br/></TrendlineDirective><br/></ChartComponent>|
| Animation settings for trendLines| Not Applicable| Property: trendLines.animation
<br/><br/><ChartComponent id='charts'><br/><TrendlineDirective animation={ enable: true
}><br/></TrendlineDirective><br/></ChartComponent>|
| Marker settings for trendLines| Not Applicable| Property: trendLines.marker
<br/><br/><ChartComponent id='charts'><br/><TrendlineDirective marker={ visible:
true}><br/></TrendlineDirective><br/></ChartComponent>|
| Tooltip for trendLines| Property: trendLines.tooltip <br/><br/>var series =[{trendlines: [{ tooltip: { } }]}]
><br/><EJ.Chart><br/>series={series}> <br/></EJ.Chart>| Property: trendLines.enableTooltip
<br/><br/><ChartComponent id='charts'><br/><TrendlineDirective
enableTooltip={true}><br/></TrendlineDirective><br/></ChartComponent>|
| DashArray for trendLines| Property: trendLines.dashArray <br/><br/>var series =[{trendlines: [{
dashArray: '10, 5' }]}]><br/><EJ.Chart><br/>series={series}> <br/></EJ.Chart>| Not Applicable. |
| Visible on legend for trendLines| Property: trendLines.visibleOnLegend <br/><br/>var series
=[{trendlines: [{ visibleOnLegend: true }]}]><br/><EJ.Chart><br/>series={series}> <br/></EJ.Chart>| Not
Applicable. |

```

Striplines

<!-- markdownlint-disable MD033 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| Default behaviour for striplines | **Property:** *primaryXAxis.striplines*

var primaryYAxis = {
stripline: [{ visible: true, start: 30, end: 40,}];
<EJ.Chart
primaryXAxis={primaryXAxis}>

</EJ.Chart> | **Property:** *primaryXAxis.striplines*

<ChartComponent id='charts'
primaryXAxis={ striplines:[{ visible: true }]}>
</ChartComponent> |

| border for stripline | **Property:** *striplines.borderColor*

var primaryYAxis = { stripline: [{
borderColor: 'pink'}];
<EJ.Chart
primaryXAxis={primaryXAxis}>

</EJ.Chart> | **Property:** *striplines.border*

<ChartComponent id='charts'
primaryXAxis={ striplines:[{border: { color: 'red' } }]}>
</ChartComponent> |

| Background color for stripline | **Property:** *striplines.color*

var primaryYAxis = { stripline: [{
color: 'pink'}];
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart> | **Property:**
striplines.color

<ChartComponent id='charts' primaryXAxis={ striplines:[{ color: 'red'
}]}>
</ChartComponent> |

| Start value for stripline | **Property:** *striplines.start*

var primaryYAxis = { stripline: [{ start:
10}];
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart> | **Property:**
striplines.start

<ChartComponent id='charts' primaryXAxis={ striplines:[{ start: 5 }
]}>
</ChartComponent> |

| End value for stripline | **Property:** *striplines.end*

var primaryYAxis = { stripline: [{ end:
10}];
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart> | **Property:** *striplines.end*

<ChartComponent id='charts' primaryXAxis={ striplines:[{ end: 5 }
]}>
</ChartComponent> |

| StartfromAxis for stripline | **Property:** *striplines.startFromAxis*

var primaryYAxis = { stripline:
[{ startFromAxis: true}];
<EJ.Chart
primaryXAxis={primaryXAxis}>

</EJ.Chart> | **Property:** *striplines.startFromAxis*

<ChartComponent id='charts'
primaryXAxis={ striplines:[{ startFromAxis: true }]}>
</ChartComponent> |

| Text in stripline | **Property:** *striplines.text*

var primaryYAxis = { stripline: [{ text:
'StripLine'}];
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart> | **Property:**
striplines.text

<ChartComponent id='charts' primaryXAxis={ striplines:[{ text:
'stripline' }]}>
</ChartComponent> |

| Text alignment in stripline | **Property:** *striplines.textAlignment*

var primaryYAxis = { stripline:
[{ textAlignment: 'Far'}];
<EJ.Chart
primaryXAxis={primaryXAxis}>

</EJ.Chart> | **Property:** *striplines.horizontalAlignment*

<ChartComponent id='charts'
primaryXAxis={ striplines:[{ horizontalAlignment: 'Far' }]}>
</ChartComponent> |

| Text alignment in stripline | Not Applicable | **Property:** *striplines.verticalAlignment*

<ChartComponent id='charts' primaryXAxis={ striplines:[{ verticalAlignment: 'Far' }
]}>
</ChartComponent> |

| Size of stripline | **Property:** *striplines.width*

var primaryYAxis = { stripline: [{ width:
10}];
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart> | **Property:** *striplines.size*

<ChartComponent id='charts' primaryXAxis={ striplines:[{ size: 10 }
]}>
</ChartComponent> |

|ZIndex of stripline| **Property:** *stripLines.zIndex*

var primaryYAxis = { stripLine: [{ zIndex: 'Behind'}]};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *stripLines.zIndex*

<ChartComponent id='charts' primaryXAxis={ stripLines:[{ zIndex: 'Behind' }]}>
</ChartComponent>|

|Font style of stripline| **Property:** *stripLines.fontStyle*

var primaryYAxis = { stripLine: [{ fontStyle: {} }]};
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *stripLines.textStyle*

<ChartComponent id='charts' primaryXAxis={ stripLines:[{ textStyle: {} }]}>
</ChartComponent>|

Multilevel Labels

<!-- markdownlint-disable MD033 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

|Default behaviour for multilevelLabels| **Property:** *stripLines.fontStyle*

var primaryYAxis = { multiLevelLabels: [{ }] };
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *primaryXAxis.multilevelLabels*

<ChartComponent id='charts' primaryXAxis={ multiLevelLabels: [{ }]}>
</ChartComponent>|

|Text alignment for multilevelLabels| **Property:** *multiLevelLabels.textAlignment*

var primaryYAxis = { multiLevelLabels: [{ textAlignment: 'Near' }] };
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *primaryXAxis.multilevelLabels*

<ChartComponent id='charts' primaryXAxis={ multiLevelLabels: [{ }]}>
</ChartComponent>|

|multilevelLabels.alignment| **Property:** *multiLevelLabels.textAlignment*

var primaryYAxis = { multiLevelLabels: [{ textAlignment: 'Near' }] };
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *multilevelLabels.alignment*

<ChartComponent id='charts' primaryXAxis={ multiLevelLabels: [{ alignment: 'Near' }]}>
</ChartComponent>|

|Text overflow for multilevelLabels| **Property:** *multiLevelLabels.textOverFlow*

var primaryYAxis = { multiLevelLabels: [{ textOverFlow : 'Trim' }] };
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *multiLevelLabels.overFlow*

<ChartComponent id='charts' primaryXAxis={ multiLevelLabels: [{ overFlow: 'Trim' }]}>
</ChartComponent>|

|Border for multilevelLabels| **Property:** *multiLevelLabels.border*

var primaryYAxis = { multiLevelLabels: [{ border: { } }] };
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *multiLevelLabels.border*

<ChartComponent id='charts' primaryXAxis={ multiLevelLabels: [{ border: { } }]}>
</ChartComponent>|

|Start value for label| **Property:** *multiLevelLabels.start*

var primaryYAxis = { multiLevelLabels: [{ start: 45 }] };
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:** *multiLevelLabels.categories.start*

<ChartComponent id='charts' primaryXAxis={ multiLevelLabels: [{ categories: [{ start: 45}] }]}>
</ChartComponent>|

|End value for label| **Property:** *multiLevelLabels.end*

var primaryYAxis = { multiLevelLabels: [{ end: 45 }] };
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:**

multiLevelLabels.categories.end

<ChartComponent id='charts' primaryXAxis={
multiLevelLabels: [{ categories: [{ end: 45}] }] }>
</ChartComponent>|

|Text for label| **Property:** *multiLevelLabels.text*

var primaryYAxis = { multiLevelLabels: [{ text:
'Start' }] };
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| **Property:**
multiLevelLabels.categories.text

<ChartComponent id='charts' primaryXAxis={
multiLevelLabels: [{ categories: [{ text: 'text' }] }] }>
</ChartComponent>|

|maximum text width for label| **Property:** *multiLevelLabels.maximumTextWidth*

var
primaryYAxis = { multiLevelLabels: [{ maximumTextWidth: 10 }] };
<EJ.Chart
primaryXAxis
={primaryXAxis}>
</EJ.Chart>| **Property:** *multiLevelLabels.categories.maximumTextWidth*

<ChartComponent id='charts' primaryXAxis={ multiLevelLabels: [{ categories: [{
maximumTextWidth: 20 }] }] }>
</ChartComponent>|

|level of label| **Property:** *multiLevelLabels.level*

var primaryYAxis = { multiLevelLabels: [{
level: 10 }] };
<EJ.Chart
primaryXAxis={primaryXAxis}>
</EJ.Chart>| Not applicable|

Methods

<!-- markdownlint-disable MD033 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

|animation for series| **Property:** *animate()*

var chartobj=
document.getElementById('chart');
<EJ.Chart
id= chart>

</EJ.Chart>
chartobj.animate();| Not applicable|

|Redraw for chart| **Property:** *redraw()*

var chartobj=
document.getElementById('chart');
<EJ.Chart
id= chart>

</EJ.Chart>
chartobj.redraw();| **Property:** *refresh()*

public loaded(args:
ILoadedEventArgs): void {
function () => {
args.chart.refresh();
<ChartComponent
id='charts'>
</ChartComponent>|

|Export| **Property:** *chart.export()*

var chartObj =
\$("#chartcontainer").ejChart("instance");
chartObj.export("chartcontainer");
<EJ.Chart
i
d= chartcontainer>
</EJ.Chart>| **Property:** *chart.export()*

public chartInstance:
ChartComponent;
public clickHandler() {
this.chartInstance.export();
<ChartComponent
id='charts'>
</ChartComponent>|

|Print| **Property:** *chart.print()*

var chartObj =
\$("#chartcontainer").ejChart("instance");
chartObj.print("chartcontainer");
<EJ.Chart
id=
chartcontainer>
</EJ.Chart>| **Property:** *chart.print()*

public chartInstance:
ChartComponent;
public clickHandler() {
this.chartInstance.print();
<ChartComponent
id='charts'>
</ChartComponent>|

|AddSeries| Not Applicable| **Property:** *chart.addSeries()*

public add()
{
this.chartInstance.addSeries([{}]) };
<ChartComponent
id='charts'>
</ChartComponent>|

|AddSeries| Not Applicable| **Property:** *chart.removeSeries()*

public add()

this.chartInstance.removeSeries([{}]);
<ChartComponent
id='charts'>
</ChartComponent>|

Events

<!-- markdownlint-disable MD033 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| Fires on annotation click| **Property:** *annotationClick*

<EJ.Chart
id= chartcontainer
annotationClick = {annotationClick}>
</EJ.Chart>
function annotationClick({});| Not
applicable|

| Fires after animation| **Property:** *animationComplete()*

<EJ.Chart
id= chartcontainer
animationComplete = {animationComplete}>
</EJ.Chart>
function animationComplete({
});| **Property:** *animationComplete()*

<ChartComponent
id='charts
animationComplete={this.animationComplete.bind(this)}'>
</ChartComponent>
public
animationComplete(args:IAnimationCompleteEventArgs): void {}|

| Fires on axis label click| **Property:** *axisLabelClick*

<EJ.Chart
id= chartcontainer
axisLabelClick = {axisLabelClick}>
</EJ.Chart>
function axisLabelClick({});| Not applicable|

| Fires before axis label render| **Property:** *axisLabelRendering*

<EJ.Chart
id=
chartcontainer axisLabelRendering = {axisLabelRendering}>
</EJ.Chart>
function
axisLabelRendering({});| **Property:** *axisLabelRender()*

<ChartComponent
id='charts
axisLabelRender={this.axisLabelRender.bind(this)}'>
</ChartComponent>
public
axisLabelRender(): void {}|

| Fires on axis label mouseMove| **Property:** *axisLabelMouseMove*

<EJ.Chart
id=
chartcontainer axisLabelMouseMove = {axisLabelMouseMove}>
</EJ.Chart>
function
axisLabelRendering({});| Not applicable|

| Fires on axis label initialize| **Property:** *axisLabelInitialize*

<EJ.Chart
id= chartcontainer
axisLabelInitialize = {axisLabelInitialize}>
</EJ.Chart>
function axisLabelInitialize({});| Not
applicable|

| Fires before axis range calculation| **Property:** *axesRangeCalculate*

<EJ.Chart
id=
chartcontainer axesRangeCalculate = {axesRangeCalculate}>
</EJ.Chart>
function
axesRangeCalculate({});| **Property:** *axisRangeCalculated*

<ChartComponent
id='charts
axisRangeCalculated={this.axisRangeCalculated.bind(this)}>
</ChartComponent>
public
axisRangeCalculated(): void {}|

| Fires on axis title rendering| **Property:** *axisTitleRendering*

<EJ.Chart
id=
chartcontainer axisTitleRendering = {axisTitleRendering}>
</EJ.Chart>
function
axisTitleRendering({});| Not applicable|

| Fires on after chart resize| **Property:** *afterResize*

<EJ.Chart
id= chartcontainer
afterResize = {afterResize}>
</EJ.Chart>
function afterResize({});| Not applicable|

| Fires on before chart resize | **Property:** *beforeResize*

<EJ.Chart
id= chartcontainer
beforeResize = {beforeResize}>
</EJ.Chart>
function beforeResize(){ }; | **Property:** *resized*

<ChartComponent
id='charts
resized={this.resized.bind(this)}>
</ChartComponent>
public resized(): void {};

| Fires on chart click | **Property:** *chartClick*

<EJ.Chart
id= chartcontainer chartClick=
{chartClick}>
</EJ.Chart>
function chartClick(){ }; | **Property:** *chartMouseClicked*

<ChartComponent
id='charts
chartMouseClicked={this.chartMouseClicked.bind(this)}>
</ChartComponent>
public
chartMouseClicked(): void {};

| Fires on chart mouse move | **Property:** *chartMouseMove*

<EJ.Chart
id=
chartcontainer chartMouseMove= {chartMouseMove}>
</EJ.Chart>
function
chartMouseMove(){ }; | **Property:** *chartMouseMove*

<ChartComponent
id='charts
chartMouseMove={this.chartMouseMove.bind(this)}>
</ChartComponent>
public
chartMouseMove(): void {};

| Fires on chart mouse leave | **Property:** *chartMouseLeave*

<EJ.Chart
id=
chartcontainer chartMouseLeave= {chartMouseLeave}>
</EJ.Chart>
function
chartMouseLeave(){ }; | **Property:** *chartMouseLeave*

<ChartComponent
id='charts
chartMouseLeave={this.chartMouseLeave.bind(this)}>
</ChartComponent>
public
chartMouseLeave(): void {};

| Fires on before chart double click | **Property:** *chartDoubleClick*

<EJ.Chart
id=
chartcontainer chartDoubleClick= {chartDoubleClick}>
</EJ.Chart>
function
chartDoubleClick(){ }; | Not applicable |

| Fires on chart mouse up | Not Applicable | **Property:** *chartmouseUp*

<ChartComponent
id='charts
chartmouseUp={this.chartmouseUp.bind(this)}>
</ChartComponent>
public
chartmouseUp(): void {};

| Fires on chart mouse down | Not Applicable | **Property:** *chartmouseDown*

<ChartComponent
id='charts
chartmouseDown={this.chartmouseDown.bind(this)}>
</ChartComponent>
public
chartmouseDown(): void {};

| Fires during the calculation of chart area bounds. You can use this event to customize the bounds of
chart area | **Property:** *chartAreaBoundsCalculate*

<EJ.Chart
id= chartcontainer
chartAreaBoundsCalculate= {chartAreaBoundsCalculate}>
</EJ.Chart>
function
chartAreaBoundsCalculate(){ }; | Not applicable |

| Fires when the dragging is started | **Property:** *dragStart*

<EJ.Chart
id= chartcontainer
dragStart= {dragStart}>
</EJ.Chart>
function dragStart(){ }; | Not applicable |

| Fires while dragging | **Property:** *dragging*

<EJ.Chart
id= chartcontainer dragging=
{dragging}>
</EJ.Chart>
function dragging(){ }; | Not applicable |

| Fires when the dragging is completed | **Property:** *dragEnd*

<EJ.Chart
id=
chartcontainer dragEnd= {dragEnd}>
</EJ.Chart>
function dragEnd(){ }; | **Property:**

```

dragComplete <br/><br/><ChartComponent<br/>id='charts
dragComplete={this.dragComplete.bind(this)}><br/></ChartComponent><br/>public
dragComplete(): void {}|

| Fires when chart is destroyed completely| Property: destroy <br/><br/><EJ.Chart<br/>id=
chartcontainer destroy= {destroy}><br/></EJ.Chart><br/>function destroy(){ };| Not applicable|

| Fires after chart is created.| Property: create <br/><br/><EJ.Chart<br/>id= chartcontainer create=
{create}><br/></EJ.Chart><br/>function create(){ };| Property: loaded
<br/><br/><ChartComponent<br/>id='charts
loaded={this.loaded.bind(this)}><br/></ChartComponent><br/>public loaded(): void {}|

| Fires before rendering the data labels.| Property: displayTextRendering <br/><br/><EJ.Chart<br/>id=
chartcontainer displayTextRendering= {displayTextRendering}><br/></EJ.Chart><br/>function
displayTextRendering(){ };| Property: textRender <br/><br/><ChartComponent<br/>id='charts
textRender={this.textRender.bind(this)}><br/></ChartComponent><br/>public textRender(): void
{}|

| Fires, when error bar is rendering| Property: errorBarRendering <br/><br/><EJ.Chart<br/>id=
chartcontainer errorBarRendering= {errorBarRendering}><br/></EJ.Chart><br/>function
errorBarRendering(){ };| Not applicable|

| Fires during the calculation of legend bounds| Property: legendBoundsCalculate
<br/><br/><EJ.Chart<br/>id= chartcontainer legendBoundsCalculate=
{legendBoundsCalculate}><br/></EJ.Chart><br/>function legendBoundsCalculate(){ };| Not
applicable|

| Fires on clicking the legend item.| Property: legendItemClick <br/><br/><EJ.Chart<br/>id=
chartcontainer legendItemClick= {legendItemClick}><br/></EJ.Chart><br/>function
legendItemClick(){ };| Not applicable|

| Fires when moving mouse over legend item| Property: legendItemMouseMove
<br/><br/><EJ.Chart<br/>id= chartcontainer legendItemMouseMove=
{legendItemMouseMove}><br/></EJ.Chart><br/>function legendItemMouseMove(){ };| Not
applicable|

| Fires before rendering the legend item.| Property: legendItemRendering <br/><br/><EJ.Chart<br/>id=
chartcontainer legendItemRendering= {legendItemRendering}><br/></EJ.Chart><br/>function
legendItemRendering(){ };| Property: legendRender <br/><br/><ChartComponent<br/>id='charts
legendRender={this.legendRender.bind(this)}><br/></ChartComponent><br/>public
legendRender(): void {}|

| Fires before loading the chart.| Property: load <br/><br/><EJ.Chart<br/>id= chartcontainer load=
{load}><br/></EJ.Chart><br/>function load(){ };| Property: load
<br/><br/><ChartComponent<br/>id='charts
load={this.load.bind(this)}><br/></ChartComponent><br/>public load(): void {}|

| Fires, when multi level labels are rendering.| Property: multiLevelLabelRendering
<br/><br/><EJ.Chart<br/>id= chartcontainer multiLevelLabelRendering=
{multiLevelLabelRendering}><br/></EJ.Chart><br/>function multiLevelLabelRendering(){

```



```

};| Property: axisMultiLabelRender <br/><br/><ChartComponent<br/>id='charts'
axisMultiLabelRender={this.axisMultiLabelRender.bind(this)}><br/></ChartComponent><br/>pu
blic axisMultiLabelRender(): void {};}

| Fires on clicking a point in chart. | Property: pointRegionClick <br/><br/><EJ.Chart<br/>id=
'chartcontainer' pointRegionClick= {pointRegionClick}><br/></EJ.Chart><br/>function
pointRegionClick(){ };| Property: pointClick <br/><br/><ChartComponent<br/>id='charts'
pointClick={this.pointClick.bind(this)}><br/></ChartComponent><br/>public pointClick(): void {};}

| Fires when mouse is moved over a point. | Property: pointRegionMouseMove
<br/><br/><EJ.Chart<br/>id= 'chartcontainer' pointRegionMouseMove=
{pointRegionMouseMove}><br/></EJ.Chart><br/>function pointRegionMouseMove(){ };| Property:
pointMove <br/><br/><ChartComponent<br/>id='charts'
pointMove={this.pointMove.bind(this)}><br/></ChartComponent><br/>public pointMove(): void
{};}

| Fires before rendering chart. | Property: preRender <br/><br/><EJ.Chart<br/>id= 'chartcontainer'
preRender= {preRender}><br/></EJ.Chart><br/>function preRender(){ };| Not applicable|

| Fires when point render. | Not Applicable| Property: pointRender
<br/><br/><ChartComponent<br/>id='charts'
pointRender={this.pointRender.bind(this)}><br/></ChartComponent><br/>public pointRender():
void {};}

| Fires after selected the data in chart | Property: rangeSelected <br/><br/><EJ.Chart<br/>id=
'chartcontainer' rangeSelected= {rangeSelected}><br/></EJ.Chart><br/>function rangeSelected(){
};| Not applicable|

| Fires after selecting a series. | Property: seriesRegionClick <br/><br/><EJ.Chart<br/>id=
'chartcontainer' seriesRegionClick= {seriesRegionClick}><br/></EJ.Chart><br/>function
seriesRegionClick(){ };| Not applicable|

| Fires before rendering a series. | Property: seriesRendering <br/><br/><EJ.Chart<br/>id=
'chartcontainer' seriesRendering= {seriesRendering}><br/></EJ.Chart><br/>function
seriesRendering(){ };| Property: seriesRender <br/><br/><ChartComponent<br/>id='charts'
seriesRender={this.seriesRender.bind(this)}><br/></ChartComponent><br/>public seriesRender():
void {};}

| Fires before rendering the marker symbols. | Property: symbolRendering <br/><br/><EJ.Chart<br/>id=
'chartcontainer' symbolRendering= {symbolRendering}><br/></EJ.Chart><br/>function
symbolRendering(){ };| Not applicable|

| Fires before rendering the trendline | Property: trendlineRendering <br/><br/><EJ.Chart<br/>id=
'chartcontainer' trendlineRendering= {trendlineRendering}><br/></EJ.Chart><br/>function
trendlineRendering(){ };| Not applicable|

| Fires before rendering the Chart title. | Property: titleRendering <br/><br/><EJ.Chart<br/>id=
'chartcontainer' titleRendering= {titleRendering}><br/></EJ.Chart><br/>function titleRendering(){
};| Not applicable|

```


| Fires before rendering the Chart sub title. | **Property:** *subTitleRendering*

<EJ.Chart
id='chartcontainer' subTitleRendering= {subTitleRendering}>
</EJ.Chart>
function subTitleRendering(){ }; | Not applicable |

| Fires before rendering the tooltip. | **Property:** *tooltipInitialize*

<EJ.Chart
id='chartcontainer' tooltipInitialize= {tooltipInitialize}>
</EJ.Chart>
function tooltipInitialize(){ }; | **Property:** *tooltipRender*

<ChartComponent
id='charts' tooltipRender={this.tooltipRender.bind(this)}>
</ChartComponent>
public tooltipRender(): void {}; |

| Fires before rendering crosshair tooltip in axis | **Property:** *trackAxisToolTip*

<EJ.Chart
id='chartcontainer' trackAxisToolTip= {trackAxisToolTip}>
</EJ.Chart>
function trackAxisToolTip(){ }; | Not applicable |

| Fires before rendering trackball tooltip. | **Property:** *trackToolTip*

<EJ.Chart
id='chartcontainer' trackToolTip= {trackToolTip}>
</EJ.Chart>
function trackToolTip(){ }; | Not applicable |

| Event triggered when scroll starts | **Property:** *scrollStart*

<EJ.Chart
id='chartcontainer' scrollStart= {scrollStart}>
</EJ.Chart>
function scrollStart(){ }; | **Property:** *scrollStart*

<ChartComponent
id='charts' scrollStart={this.scrollStart.bind(this)}>
</ChartComponent>
public scrollStart(): void {}; |

| Event triggered when scroll ends. | **Property:** *scrollEnd*

<EJ.Chart
id='chartcontainer' scrollEnd= {scrollEnd}>
</EJ.Chart>
function scrollEnd(){ }; | **Property:** *scrollEnd*

<ChartComponent
id='charts' scrollEnd={this.scrollEnd.bind(this)}>
</ChartComponent>
public scrollEnd(): void {}; |

| Event triggered when scroll Change. | **Property:** *scrollChange*

<EJ.Chart
id='chartcontainer' scrollChange= {scrollChange}>
</EJ.Chart>
function scrollChange(){ }; | **Property:** *scrollChange*

<ChartComponent
id='charts' scrollChange={this.scrollChange.bind(this)}>
</ChartComponent>
public scrollChange(): void {}; |

| Fires while performing rectangle zooming in chart | **Property:** *zoomComplete*

<EJ.Chart
id='chartcontainer' zoomComplete= {zoomComplete}>
</EJ.Chart>
function zoomComplete(){ }; | **Property:** *zoomComplete*

<ChartComponent
id='charts' zoomComplete={this.zoomComplete.bind(this)}>
</ChartComponent>
public zoomComplete(): void {}; |

Chart properties

<!-- markdownlint-disable MD033 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| selected data index | **Property:** *selectedDataPointIndexes*

var selectedDataPointIndexes = [{ seriesIndex: 0, pointIndex: 1}];

<EJ.Chart
id='chartcontainer' selectedDataPointIndexes= {selectedDataPointIndexes}>
</EJ.Chart> | **Property:** *selectedDataIndexes*

public selectedDataIndexes: [{ series: 0, point: 1}];

<ChartComponent
id='charts' selectedDataIndexes={this.selectedDataIndexes}>
</ChartComponent> |

|sideBySideSeriesPlacement for column based series| **Property:** *sideBySideSeriesPlacement*
`

var sideBySideSeriesPlacement = [{ seriesIndex: 0, pointIndex: 1}];
<EJ.Chart
id= 'chartcontainer' sideBySideSeriesPlacement= {sideBySideSeriesPlacement}>
</EJ.Chart>| Property: sideBySidePlacement

public sideBySidePlacement: [{true}];
<ChartComponent
id='charts' sideBySidePlacement={this.sideBySidePlacement}>
</ChartComponent>
|`

|ZoomSettings| **Property:** *zooming*

var zooming = {enable: true};
<EJ.Chart
id= 'chartcontainer' zooming= {zooming}>
</EJ.Chart>| **Property:** *zoomSettings*

public zoomSettings: ZoomSettingsModel =
{enableMouseWheelZooming: true}
<ChartComponent
id='charts' zoomSettings={this.zoomSettings}>
</ChartComponent>
|

|Background color of the chart| **Property:** *background*

<EJ.Chart
id= 'chartcontainer' background='transparent'>
</EJ.Chart>| **Property:** *background*

<ChartComponent
id='charts' background='#EEFFCC'>
</ChartComponent>
|

|URL of the image to be used as chart background. | **Property:** *backGroundImageUrl*

<EJ.Chart
id= 'chartcontainer' background='transparent'>
</EJ.Chart>| Not Applicable|

|Customizing border of the chart| **Property:** *border*

<EJ.Chart
id= 'chartcontainer' border= {}>
</EJ.Chart>| **Property:** *border*

<ChartComponent
id='charts' border= {}>
</ChartComponent>
|

|This provides options for customizing export settings| **Property:** *exportSettings*

var exportSettings= { filename : "chart", angle: '45' }
<EJ.Chart
id= 'chartcontainer' exportSettings={exportSettings}>
</EJ.Chart>| **Property:** *export()*

<ChartComponent
id='charts' border= {}>
</ChartComponent>
public onClick(): void
let fileName: string = (document.getElementById('fileName') as HTMLInputElement).value;
 this.chartInstance.export((this.mode.value as ExportType), fileName);|

|ChartArea customization| **Property:** *chartArea*

var chartArea= {}
<EJ.Chart
id= 'chartcontainer' chartArea={chartArea}>
</EJ.Chart>| **Property:** *chartArea*

<ChartComponent
id='charts' chartArea={this.chartArea}>
</ChartComponent>|

How To

Live chart in React Chart component

You can update a chart with live data by using the set interval.

To update live data in a chart, follow the given steps:

Step 1:

Initialize the chart with series.

Step 2:

Update the data to series using set interval, and shift the data in the series to make the series look like moving.

INDEX.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
LineSeries } from '@syncfusion/ej2-react-charts';
function App() {
    var chart;
    var intervalId;
    var series1 = [];
    var value = 10;
    var setTimeoutValue = 100;
    for (var i = 0; i < 50; i++) {
        if (Math.random() > 0.5) {
            value += Math.random() * 2.0;
        }
        series1[i] = { x: i, y: value };
    }
    chart = chart;
    function loaded(args) {
        intervalId = setTimeout(() => {
            if (chart === null) {
                clearInterval(intervalId);
            }
            else {
                if (Math.random() > 0.5) {
                    value += Math.random() * 2.0;
                }
                i++;
                series1.push({ x: i, y: value });
                series1.shift();
                args.chart.series[0].dataSource = series1;
            }
        }, setTimeoutValue);
    }
    return (<ChartComponent id='charts' loaded={loaded.bind(this)}>
        <Inject services={[LineSeries]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={series1} xName='x' yName='y'
type='Line'></SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
```

INDEX.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, LineSeries, ILoadedEventArgs }
from '@syncfusion/ej2-react-charts';
```

```

import { getElement } from '@syncfusion/ej2-charts';
function App() {
  var chart;
  var intervalId;
  var series1 = [];
  var value = 10;
  var setTimeoutValue = 100;
  for (var i = 0; i < 50; i++) {
    if (Math.random() > 0.5) {
      value += Math.random() * 2.0;
    }
    series1[i] = { x: i, y: value };
  }
  chart = chart;
  function loaded(args) {
    intervalId = setTimeout(() => {
      if (chart === null) {
        clearInterval(intervalId);
      } else {
        if (Math.random() > 0.5) {
          value += Math.random() * 2.0;
        }
        i++;
        series1.push({ x: i, y: value });
        series1.shift();
        args.chart.series[0].dataSource = series1;
      }
    }, setTimeoutValue);
  }
  return (
    <ChartComponent id='charts' loaded={loaded.bind(this)}>
      <Inject services={[LineSeries]} />
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={series1} xName='x' yName='y'
type='Line'></SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));

```

Prevent data label in React Chart component

To prevent the chart data label when the data value is 0, follow the given steps:

Step 1:

Get the point value and check whether the `args.point.y` value is zero or not by using the [textRender](#) event. If the value is zero, then set the `args.cancel` to true.

The output will appear as follows,

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";

```

```

import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
function App() {
    const data = [
        { x: new Date(2005, 0, 1), y: 21 }, { x: new Date(2006, 0, 1), y: 24
    },
        { x: new Date(2007, 0, 1), y: 0 }, { x: new Date(2008, 0, 1), y: 38
    },
        { x: new Date(2009, 0, 1), y: 54 }, { x: new Date(2010, 0, 1), y: 57
    },
    ];
    const primaryxAxis = { valueType: 'DateTime' };
    const primaryyAxis = { minimum: 0, maximum: 100, interval: 10 };
    const marker = { visible: true, dataLabel: { visible: true } };
    const textRender = (args) => {
        if (args.text === '0') {
            args.cancel = args.point.y === 0;
        }
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Inflation - Consumer Price'
textRender={textRender}>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel,
DateTime]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='Germany' type='Line' marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, LineSeries,
ITextRenderEventArgs, AxisModel } from '@syncfusion/ej2-react-charts';
import { EmitType } from '@syncfusion/ej2-base'
function App() {
    const data: any[] = [
        { x: new Date(2005, 0, 1), y: 21 }, { x: new Date(2006, 0, 1), y: 24 },
        { x: new Date(2007, 0, 1), y: 0 }, { x: new Date(2008, 0, 1), y: 38 },
        { x: new Date(2009, 0, 1), y: 54 }, { x: new Date(2010, 0, 1), y: 57 },
    ];
    const primaryxAxis: AxisModel = { valueType: 'DateTime' };
    const primaryyAxis: AxisModel = { minimum: 0, maximum: 100, interval: 10
    };
    const marker = { visible: true, dataLabel: { visible: true } };
    const textRender: EmitType<ITextRenderEventArgs> = (args:
ITextRenderEventArgs): void => {

```

```

    if (args.text === '0') {
      args.cancel = args.point.y === 0;
    }
  };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Inflation - Consumer Price'
    textRender={textRender}>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, DateTime]}
  />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='Germany'
        type='Line' marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Tool tip format in React Chart component

Using [tooltipRender](#) event, you can able to format the datetime value instead of rendered value.

To format the datetime value, please follow the steps below

Step 1:

By using [tooltipRender](#) event we can able to get the current point x value. Using this value to format the tooltip by using `formatDate` method.

The output will appear as follows,

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
import { Internationalization } from '@syncfusion/ej2-base';
function App() {
  const data = [
    { x: new Date(2005, 0, 1), y: 21 }, { x: new Date(2006, 0, 1), y: 24
  },
    { x: new Date(2007, 0, 1), y: 30 }, { x: new Date(2008, 0, 1), y: 38
  },
    { x: new Date(2009, 0, 1), y: 54 }, { x: new Date(2010, 0, 1), y: 57
  },
  ];
  const primaryxAxis = { valueType: 'DateTime' };
  const marker = { visible: true, height: 10, width: 10, dataLabel: {
visible: true } };
  const tooltip = { enable: true };
  const tooltipRender = (args) => {
    let intl = new Internationalization();

```

```

        let formattedString = intl.formatDate(new Date(args.point.x), {
skeleton: 'yMd' });
        args.text = formattedString;
    };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
title='Inflation - Consumer Price' tooltipRender={tooltipRender}
tooltip={tooltip}>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel,
DateTime]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='Germany' type='Line' marker={marker}>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
    }
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, DateTime, Tooltip, DataLabel, LineSeries, ITextRenderEventArgs,
ITooltipRenderEventArgs, AxisModel, TooltipSettingsModel
} from '@syncfusion/ej2-react-charts';
import { Internationalization } from '@syncfusion/ej2-base';
function App() {
    const data: any[] = [
        { x: new Date(2005, 0, 1), y: 21 }, { x: new Date(2006, 0, 1), y: 24 },
        { x: new Date(2007, 0, 1), y: 30 }, { x: new Date(2008, 0, 1), y: 38 },
        { x: new Date(2009, 0, 1), y: 54 }, { x: new Date(2010, 0, 1), y: 57 },
    ];
    const primaryxAxis: AxisModel = { valueType: 'DateTime' };
    const marker = { visible: true, height: 10, width: 10, dataLabel: {
visible: true } };
    const tooltip: TooltipSettingsModel = { enable: true };
    const tooltipRender = (args: ITooltipRenderEventArgs) => {
        let intl: Internationalization = new Internationalization();
        let formattedString: string = intl.formatDate(new Date(args.point.x), {
skeleton: 'yMd' });
        args.text = formattedString;
    };
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        title='Inflation - Consumer Price'
        tooltipRender={tooltipRender}
        tooltip={tooltip}>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel, DateTime]}
/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='Germany'
                type='Line' marker={marker}>

```

```

    </SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Add series in React Chart component

You can add or remove the chart series dynamically by using the `addSeries` or `removeSeries` method.

To add or remove the series dynamically, follow the given steps:

Step 1:

To add a new series to chart dynamically, pass the series value to the `addSeries` method.

To remove the new series from chart dynamically, pass the series index to the `removeSeries` method.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, LineSeries } from
 '@syncfusion/ej2-react-charts';
function App() {
  const data = [{ x: 'John', y: 10000 },
    { x: 'Jake', y: 12000 },
    { x: 'Peter', y: 18000 },
    { x: 'James', y: 11000 },
  ];
  var chartInstance;
  function add() {
    chartInstance.addSeries([
      {
        type: 'Column',
        dataSource: [
          { x: 'John', y: 11000 }, { x: 'Jake', y: 16000 },
          { x: 'Peter', y: 19000 },
          { x: 'James', y: 12000 }, { x: 'Mary', y: 10700 }
        ],
        xName: 'x', width: 2,
        yName: 'y'
      }
    ]);
  }
  ;
  function remove() {
    chartInstance.removeSeries(1);
  }
  ;
  return (
    <div>
      <ButtonComponent value='add' cssClass='e-flat'
        onClick={add.bind(this)}>add</ButtonComponent>
      <ButtonComponent value='remove' cssClass='e-flat'
        onClick={remove.bind(this)}>remove</ButtonComponent>
      <ChartComponent id='charts' ref={chart => chartInstance = chart}
        primaryXAxis={{ valueType: 'Category' }} title='Sales Comparision'>

```



```

        <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y'
type='Column' name='Sales'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent> </div>);
}
;
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject,
ColumnSeries, Legend, Category, Tooltip, DataLabel, Zoom, Crosshair,
LineSeries, Selection } from '@syncfusion/ej2-react-charts';
function App() {
    const data: any[] = [{ x: 'John', y: 10000 },
    { x: 'Jake', y: 12000 },
    { x: 'Peter', y: 18000 },
    { x: 'James', y: 11000 },
    ];
    var chartInstance: ChartComponent;
    function add() {
        chartInstance.addSeries([
            {
                type: 'Column',
                dataSource: [
                    { x: 'John', y: 11000 },
                    { x: 'Jake', y: 16000 },
                    { x: 'Peter', y: 19000 },
                    { x: 'James', y: 12000 },
                    { x: 'Mary', y: 10700 }
                ],
                xName: 'x', width: 2,
                yName: 'y'
            }
        ]);
    };
    function remove() {
        chartInstance.removeSeries(1);
    };
    return (
        <div>
            <ButtonComponent value='add' cssClass='e-flat'
onClick={add.bind(this)}>add</ButtonComponent>
            <ButtonComponent value='remove' cssClass='e-flat'
onClick={remove.bind(this)}>remove</ButtonComponent>
            <ChartComponent id='charts' ref={chart => chartInstance = chart}
primaryXAxis={{ valueType: 'Category' }}
title='Sales Comparision'>
                <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
LineSeries, Category]}/>
                <SeriesCollectionDirective>

```

```

        <SeriesDirective dataSource={data} xName='x' yName='y'
type='Column' name='Sales'>
        </SeriesDirective>
    </SeriesCollectionDirective>
</ChartComponent> </div>)
};
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Points customization in React Chart component

You can customize the series points by using the `pointColorMapping` property.

To customize the series point colors, follow the given steps:

Step 1:

Customize the point colors to set the color value by using the `pointColorMapping` property.

INDEX.JSX

```

{% raw %}
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import {
    ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
    Legend, Category, Tooltip, DataLabel, ColumnSeries,
} from '@syncfusion/ej2-react-charts';
function App() {
    const data = [
        { country: 'USA', gold: 50, color: 'url(#chess)' },
        { country: 'China', gold: 40, color: 'url(#cross)' },
        { country: 'Japan', gold: 70, color: 'url(#circle)' },
        { country: 'Australia', gold: 60, color: 'url(#chess)' },
        { country: 'France', gold: 50, color: 'url(#rectangle)' },
        { country: 'Germany', gold: 40, color: 'url(#chess)' },
        { country: 'Italy', gold: 40, color: 'url(#line)' },
        { country: 'Sweden', gold: 30, color: 'url(#cross)' }
    ];
    const primaryxAxis = { valueType: 'Category', title: 'Countries' };
    const primaryyAxis = {
        minimum: 0,
        maximum: 80,
        interval: 20,
        title: 'Medals',
    };
    return (
        <ChartComponent
            id="charts"
            primaryXAxis={primaryxAxis}
            primaryYAxis={primaryyAxis}
            title="Olympic Medals"
        >
            <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
            <SeriesCollectionDirective>
                <SeriesDirective

```

```

        dataSource={data}
        xName="country"
        yName="gold"
        name="Gold"
        type="Column"
        pointColorMapping="color"
      ></SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import {
  ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
  Legend, Category, Tooltip, DataLabel, ColumnSeries, AxisModel
} from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { country: 'USA', gold: 50, color: 'url(#chess)' },
    { country: 'China', gold: 40, color: 'url(#cross)' },
    { country: 'Japan', gold: 70, color: 'url(#circle)' },
    { country: 'Australia', gold: 60, color: 'url(#chess)' },
    { country: 'France', gold: 50, color: 'url(#rectangle)' },
    { country: 'Germany', gold: 40, color: 'url(#chess)' },
    { country: 'Italy', gold: 40, color: 'url(#line)' },
    { country: 'Sweden', gold: 30, color: 'url(#cross)' }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category', title:
'Countries' };
  const primaryyAxis: AxisModel = {
    minimum: 0,
    maximum: 80,
    interval: 20,
    title: 'Medals',
  };
  return (
    <ChartComponent
      id="charts"
      primaryXAxis={primaryxAxis}
      primaryYAxis={primaryyAxis}
      title="Olympic Medals"
    >
      <Inject services={[ColumnSeries, Legend, Tooltip, DataLabel,
Category]} />
      <SeriesCollectionDirective>
        <SeriesDirective
          dataSource={data}
          xName="country"

```

```

        yName="gold"
        name="Gold"
        type="Column"
        pointColorMapping="color"
    ></SeriesDirective>
</SeriesCollectionDirective>
</ChartComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Stacking total in React Chart component

By using the [annotation](#), you can show any element in desired view.

To show the total value in data points, follow the given steps:

Step 1:

Define annotation for each x point in chart, now change the annotation value in chart by using the [annotationRender](#) event.

In this event, assign the stacked value of the last series to the annotation to show the total value of the stacking series.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
AnnotationsDirective, AnnotationDirective, Legend, Category, Tooltip,
DataLabel, StackingColumnSeries, ChartAnnotation } from '@syncfusion/ej2-
react-charts';
function App() {
    const data = [
        { x: 'Jamesh', y0: 5, y1: 4, y2: 5 },
        { x: 'Michael', y0: 4, y1: 3, y2: 4 },
        { x: 'John', y0: 5, y1: 4, y2: 2 }
    ];
    const primaryxAxis = { valueType: 'Category' };
    const primaryyAxis = { minimum: 0, maximum: 15, interval: 5 };
    const marker1 = { visible: true, dataLabel: { position: 'Top', font: {
fontWeight: '600', color: 'ffffff' } } };
    const marker2 = { visible: true, dataLabel: { position: 'Top', font: {
fontWeight: '600', color: 'ffffff' } } };
    const marker3 = { visible: true, dataLabel: { position: 'Top', font: {
fontWeight: '600', color: 'ffffff' } } };
    const template1 = chartTemplate1;
    function chartTemplate1() {
        return (


11</span>
</div>


```

```

    </div>);
  }
;
const template2 = chartTemplate2;
function chartTemplate2() {
  return (<div className='template'>
    <div style={{ color: 'gray', fontSize: '11px', fontWeight: 'bold',
fill: 'gray' }}>
      <span>10</span>
    </div>
  </div>);
}
;
const template3 = chartTemplate3;
function chartTemplate3() {
  return (<div className='template'>
    <div style={{ color: 'gray', fontSize: '11px', fontWeight: 'bold',
fill: 'gray' }}>
      <span>12</span>
    </div>
  </div>);
}
;
return <ChartComponent id='charts' primaryXAxis={primaryXAxis}
primaryYAxis={primaryYAxis} title='Fruit Consumption'>
  <Inject services={[StackingColumnSeries, Legend, Tooltip, DataLabel,
Category, ChartAnnotation]}/>
  <AnnotationsDirective>
    <AnnotationDirective content={templatel} coordinateUnits='Point'
x='Jamesh' y={14.5}>
    </AnnotationDirective>
    <AnnotationDirective content={template2} coordinateUnits='Point'
x='Michael' y={12}>
    </AnnotationDirective>
    <AnnotationDirective content={template3} coordinateUnits='Point'
x='John' y={12}>
    </AnnotationDirective>
  </AnnotationsDirective>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} type='StackingColumn' xName='x'
yName='y0' name='Apple' marker={marker1}></SeriesDirective>
    <SeriesDirective dataSource={data} type='StackingColumn' xName='x'
yName='y1' name='Orange' marker={marker2}></SeriesDirective>
    <SeriesDirective dataSource={data} type='StackingColumn' xName='x'
yName='y2' name='Grapes' marker={marker3}></SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

INDEX.TSX

```
{% raw %}
```

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective,
Inject, AnnotationsDirective, AnnotationDirective, Legend, Category, Tooltip,
DataLabel, StackingColumnSeries, IAnnotationRenderEventArgs, ChartAnnotation
, AxisModel } from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { x: 'Jamesh', y0: 5, y1: 4, y2: 5 },
    { x: 'Michael', y0: 4, y1: 3, y2: 4 },
    { x: 'John', y0: 5, y1: 4, y2: 2 }
  ];
  const primaryxAxis: AxisModel = { valueType: 'Category' };
  const primaryyAxis: AxisModel = { minimum: 0, maximum: 15, interval: 5 };
  const marker1 = { visible: true, dataLabel: { position: 'Top', font: {
fontWeight: '600', color: 'ffffff' } } };
  const marker2 = { visible: true, dataLabel: { position: 'Top', font: {
fontWeight: '600', color: 'ffffff' } } };
  const marker3 = { visible: true, dataLabel: { position: 'Top', font: {
fontWeight: '600', color: 'ffffff' } } };
  const template1: any = chartTemplate1;
  function chartTemplate1(): any {
    return (<div className='template'>
      <div style={{ color: 'gray',fontSize:'11px',fontWeight:'bold',fill:
'gray'}}>
        <span>11</span>
      </div>
    </div>);
  };
  const template2: any = chartTemplate2;
  function chartTemplate2(): any {
    return (<div className='template'>
      <div style={{ color: 'gray',fontSize:'11px',fontWeight:'bold',fill:
'gray'}}>
        <span>10</span>
      </div>
    </div>);
  };
  const template3: any = chartTemplate3;
  function chartTemplate3(): any {
    return (<div className='template'>
      <div style={{ color: 'gray',fontSize:'11px',fontWeight:'bold',fill:
'gray'}}>
        <span>12</span>
      </div>
    </div>);
  };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='Fruit Consumption' >
    <Inject services={[StackingColumnSeries, Legend, Tooltip, DataLabel,
Category, ChartAnnotation]} />
    <AnnotationsDirective>
      <AnnotationDirective content={template1} coordinateUnits='Point'
x='Jamesh' y={14.5}>
        </AnnotationDirective>

```

```

    <AnnotationDirective content={template2} coordinateUnits='Point'
x='Michael' y={12}>
    </AnnotationDirective>
    <AnnotationDirective content={template3} coordinateUnits='Point'
x='John' y={12}>
    </AnnotationDirective>
  </AnnotationsDirective>
  <SeriesCollectionDirective>
    <SeriesDirective dataSource={data} type='StackingColumn' xName='x'
yName='y0' name='Apple'
      marker={marker1} ></SeriesDirective>
    <SeriesDirective dataSource={data} type='StackingColumn' xName='x'
yName='y1' name='Orange'
      marker={marker2} ></SeriesDirective>
    <SeriesDirective dataSource={data} type='StackingColumn' xName='x'
yName='y2' name='Grapes'
      marker={marker3} ></SeriesDirective>
  </SeriesCollectionDirective>
</ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}

```

Selected data grid in React Chart component

By using the [dragComplete](#), you can get the selected data values for range selection.

To display the selected data value, follow the given steps:

Step 1:

Get the selected data point values and display the values through grid component by using the [dragComplete](#) event.

INDEX.JSX

```

{% raw %}
import { ColumnDirective, ColumnsDirective, GridComponent } from
'@syncfusion/ej2-react-grids';
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Selection, Legend, Category, ScatterSeries } from '@syncfusion/ej2-react-
charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const data = [
    { x: 1971, y: 50 },
    { x: 1972, y: 20 },
    { x: 1973, y: 63 },
    { x: 1974, y: 81 },
    { x: 1975, y: 64 },
    { x: 1976, y: 36 },
    { x: 1977, y: 22 },
    { x: 1978, y: 78 },
    { x: 1979, y: 60 },
    { x: 1980, y: 41 },
    { x: 1981, y: 62 },
  ];

```

```

    { x: 1982, y: 56 },
    { x: 1983, y: 96 },
    { x: 1984, y: 48 },
    { x: 1985, y: 23 },
    { x: 1986, y: 54 },
    { x: 1987, y: 73 },
    { x: 1988, y: 56 },
    { x: 1989, y: 67 },
    { x: 1990, y: 79 },
    { x: 1991, y: 18 },
    { x: 1992, y: 78 },
    { x: 1993, y: 92 },
    { x: 1994, y: 43 },
    { x: 1995, y: 29 },
    { x: 1996, y: 14 },
    { x: 1997, y: 85 },
    { x: 1998, y: 24 },
    { x: 1999, y: 61 },
    { x: 2000, y: 80 },
    { x: 2001, y: 14 },
    { x: 2002, y: 34 },
    { x: 2003, y: 81 },
    { x: 2004, y: 70 },
    { x: 2005, y: 21 },
    { x: 2006, y: 70 },
    { x: 2007, y: 32 },
    { x: 2008, y: 43 },
    { x: 2009, y: 21 },
    { x: 2010, y: 63 },
    { x: 2011, y: 9 },
    { x: 2012, y: 51 },
    { x: 2013, y: 25 },
    { x: 2014, y: 96 },
    { x: 2015, y: 32 }
  ];
  const primaryxAxis = { minimum: 1970, maximum: 2016 };
  const primaryyAxis = { title: 'Sales', labelFormat: '{value}%',
interval: 25, minimum: 0, maximum: 100 };
  const marker = { shape: 'Triangle', width: 10, height: 10 };
  const legendSettings = { visible: true, toggleVisibility: false };
  const selectionMode = 'DragXY';
  var grid;
  var chart;
  function dragComplete(args) {
    grid.dataSource = args.selectedDataValues[0];
    grid.refresh();
  }
  return (<div>
    <div>
      <div>
        <ChartComponent id='charts' ref={(g) => (chart = g)}
primaryXAxis={primaryxAxis} primaryYAxis={primaryyAxis} title="Profit
Comparision of A and B" selectionMode={selectionMode}
legendSettings={legendSettings} dragComplete={dragComplete.bind(this)}>
        <Inject services={[Selection, Legend, Category,
ScatterSeries]} />
        <SeriesCollectionDirective>

```



```

        <SeriesDirective dataSource={data} xName='x'
yName='y' name='Product A' type='Scatter' marker={marker}></SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
</div>
</div>
<div>
  <GridComponent id='grid' ref={(g) => (grid = g)} height='250px'>
    <ColumnsDirective>
      <ColumnDirective field='x' headerText='x' type='string'
/>
      <ColumnDirective field='y' headerText='y' type='number'
/>
    </ColumnsDirective>
  </GridComponent>
</div>
</div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import { ColumnDirective, ColumnsDirective, GridComponent, Grid,
IDragCompleteEventArgs } from '@syncfusion/ej2-react-grids';
import { AxisModel, ChartComponent, Chart, SeriesCollectionDirective,
SeriesDirective, Inject, Selection, Legend, LegendSeriesModel, Category,
ScatterSeries } from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const data: any[] = [
    { x: 1971, y: 50 },
    { x: 1972, y: 20 },
    { x: 1973, y: 63 },
    { x: 1974, y: 81 },
    { x: 1975, y: 64 },
    { x: 1976, y: 36 },
    { x: 1977, y: 22 },
    { x: 1978, y: 78 },
    { x: 1979, y: 60 },
    { x: 1980, y: 41 },
    { x: 1981, y: 62 },
    { x: 1982, y: 56 },
    { x: 1983, y: 96 },
    { x: 1984, y: 48 },
    { x: 1985, y: 23 },
    { x: 1986, y: 54 },
    { x: 1987, y: 73 },
    { x: 1988, y: 56 },
    { x: 1989, y: 67 },
    { x: 1990, y: 79 },
    { x: 1991, y: 18 },
    { x: 1992, y: 78 },

```

```

        { x: 1993, y: 92 },
        { x: 1994, y: 43 },
        { x: 1995, y: 29 },
        { x: 1996, y: 14 },
        { x: 1997, y: 85 },
        { x: 1998, y: 24 },
        { x: 1999, y: 61 },
        { x: 2000, y: 80 },
        { x: 2001, y: 14 },
        { x: 2002, y: 34 },
        { x: 2003, y: 81 },
        { x: 2004, y: 70 },
        { x: 2005, y: 21 },
        { x: 2006, y: 70 },
        { x: 2007, y: 32 },
        { x: 2008, y: 43 },
        { x: 2009, y: 21 },
        { x: 2010, y: 63 },
        { x: 2011, y: 9 },
        { x: 2012, y: 51 },
        { x: 2013, y: 25 },
        { x: 2014, y: 96 },
        { x: 2015, y: 32 }
    ];
    const primaryxAxis: AxisModel = { minimum: 1970, maximum: 2016 };
    const primaryyAxis: AxisModel = { title: 'Sales', labelFormat:
'{{value}}%', interval: 25, minimum: 0, maximum: 100 };
    const marker = { shape: 'Triangle', width: 10, height: 10 };
    const legendSettings: LegendSeriesModel = { visible: true,
toggleVisibility: false };
    const selectionMode = 'DragXY';
    var grid: Grid | null;
    var chart: Chart | null;
    function dragComplete(args: IDragCompleteEventArgs): void {
        grid.dataSource = args.selectedDataValues[0];
        grid.refresh();
    }
    return (<div>
        <div>
            <div>
                <ChartComponent id='charts' ref={(g) => (chart = g)}
primaryXAxis={primaryxAxis} primaryYAxis={primaryyAxis} title="Profit
Comparision of A and B" selectionMode={selectionMode}
legendSettings={legendSettings} dragComplete={dragComplete.bind(this)}>
                <Inject services={[Selection, Legend, Category,
ScatterSeries]} />
                <SeriesCollectionDirective>
                    <SeriesDirective dataSource={data} xName='x'
yName='y' name='Product A' type='Scatter' marker={marker}></SeriesDirective>
                </SeriesCollectionDirective>
            </ChartComponent>
        </div>
    </div>
    <div>
        <GridComponent id='grid' ref={(g) => (grid = g)} height='250px'>
            <ColumnsDirective>

```

```

        <ColumnDirective field='x' headerText='x' type='string'
    />
        <ColumnDirective field='y' headerText='y' type='number'
    />
    </ColumnsDirective>
</GridComponent>
</div>
</div>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.HTML

```

<!DOCTYPE html>
<html lang="en">
<head>
    <title>Syncfusion React Chart-Category Axis</title>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta name="description" content="Essential JS 2 for React Components"
    />
    <meta name="author" content="Syncfusion" />
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/systemjs/0.19.38/system.js"></sc
ript>
    <script src="systemjs.config.js"></script>
    <style>
        #loader {
            color: #008cff;
            height: 40px;
            left: 45%;
            position: absolute;
            top: 45%;
            width: 30%;
        }
        #charts {
            height : 350px;
            display: block;
        }
    </style>
</head>
<body>
    <div id='charts'>
        <div id='loader'>Loading....</div>
    </div>
    <label id="lbl"></label>
    <div id='grid'></div>
</body>
</html>

```

Marker customization in React Chart component

By using the [pointRender](#), you can customize the marker shape.

To Customize the marker shape, follow the given steps:

Step 1:

Customize the marker shape in each data point by using the [pointRender](#) event.

Using this event, you can set the **shape** value to the argument.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, LineSeries } from '@syncfusion/ej2-
react-charts';
function App() {
  const data = [
    { x: 'WW', y: 12, text: 'World Wide' },
    { x: 'EU', y: 5, text: 'Europe' },
    { x: 'APAC', y: 15, text: 'Pacific' },
    { x: 'LATAM', y: 6.4, text: 'Latin' },
    { x: 'MEA', y: 30, text: 'Africa' },
    { x: 'NA', y: 25.3, text: 'America' }
  ];
  const pointRender = (args) => {
    let shapes = ['Diamond', 'Circle', 'Rectangle', 'Line', 'Triangle',
'Rectangle'];
    args.shape = shapes[args.point.index];
  };
  const primaryxAxis = { valueType: 'Category' };
  const primaryyAxis = { minimum: 0, maximum: 60, interval: 15 };
  const marker = {
    visible: true, width: 10, height: 10,
    shape: 'Diamond'
  };
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='FB Penetration of Internet Audience'
pointRender={pointRender}>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel,
Category]}>
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='December 2007' type='Line' marker={marker}>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
```

```

import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
AxisModel, Legend, Category, Tooltip, DataLabel, LineSeries, Marker,
IPointRenderEventArgs } from '@syncfusion/ej2-react-charts';
import { EmitType } from '@syncfusion/ej2-base';
function App() {
  const data: any[] = [
    { x: 'WW', y: 12, text: 'World Wide' },
    { x: 'EU', y: 5, text: 'Europe' },
    { x: 'APAC', y: 15, text: 'Pacific' },
    { x: 'LATAM', y: 6.4, text: 'Latin' },
    { x: 'MEA', y: 30, text: 'Africa' },
    { x: 'NA', y: 25.3, text: 'America' }
  ];
  const pointRender: EmitType<IPointRenderEventArgs> = (args:
IPointRenderEventArgs): void => {
    let shapes: string[] = ['Diamond', 'Circle', 'Rectangle', 'Line',
'Triangle', 'Rectangle'];
    args.shape = shapes[args.point.index];
  };
  const primaryxAxis: AxisModel = { valueType: 'Category' };
  const primaryyAxis: AxisModel = { minimum: 0, maximum: 60, interval: 15 };
  const marker = {
    visible: true, width: 10, height: 10,
    shape: 'Diamond'
  };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='FB Penetration of Internet Audience'
    pointRender={pointRender}>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category]}
  />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' width={2}
name='December 2007'
        type='Line' marker={marker} >
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Legend customization in React Chart component

By using the [legendRender](#), you can customize the legend shape.

To Customize the legend shape, follow the given steps:

Step 1:

Set the shape value for each legend using `args.shape` in

[legendRender](#) event.

INDEX.JSX

```
import * as React from "react";
```

```

import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
Legend, Category, Tooltip, DataLabel, StepAreaSeries } from
'@syncfusion/ej2-react-charts';
function App() {
    const data = [
        { x: 2000, y: 416 }, { x: 2001, y: 490 },
        { x: 2002, y: 470 }, { x: 2003, y: 500 },
        { x: 2004, y: 449 }, { x: 2005, y: 470 },
        { x: 2006, y: 437 }, { x: 2007, y: 458 }
    ];
    const data1 = [
        { x: 2000, y: 180 }, { x: 2001, y: 240 },
        { x: 2002, y: 370 }, { x: 2003, y: 200 },
        { x: 2004, y: 229 }, { x: 2005, y: 210 },
        { x: 2006, y: 337 }, { x: 2007, y: 258 }
    ];
    const legendRender = (args) => {
        if (args.text === 'Renewable') {
            args.shape = 'Circle';
        }
        else if (args.text === 'Non-Renewable') {
            args.shape = 'Triangle';
        }
    };
    const primaryxAxis = { valueType: 'Double' };
    const primaryyAxis = { valueType: 'Double' };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='FB Penetration of Internet Audience'
legendRender={legendRender}>
        <Inject services={[StepAreaSeries, Legend, Tooltip, DataLabel,
Category]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} type='StepArea' xName='x'
yName='y' width={2} name='Renewable'>
            </SeriesDirective>
            <SeriesDirective dataSource={data1} type='StepArea' xName='x'
yName='y' width={2} name='Non-Renewable'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
AxisModel, Legend, Category, Tooltip, DataLabel, StepAreaSeries ,
ILegendRenderEventArgs } from '@syncfusion/ej2-react-charts';
import { EmitType } from '@syncfusion/ej2-base';
function App() {
    const data: any[] = [

```

```

    { x: 2000, y: 416 }, { x: 2001, y: 490 },
    { x: 2002, y: 470 }, { x: 2003, y: 500 },
    { x: 2004, y: 449 }, { x: 2005, y: 470 },
    { x: 2006, y: 437 }, { x: 2007, y: 458 }
  ];
  const data1: any[] = [
    { x: 2000, y: 180 }, { x: 2001, y: 240 },
    { x: 2002, y: 370 }, { x: 2003, y: 200 },
    { x: 2004, y: 229 }, { x: 2005, y: 210 },
    { x: 2006, y: 337 }, { x: 2007, y: 258 }
  ];
  const legendRender: EmitType<ILegendRenderEventArgs> = (args:
  ILegendRenderEventArgs): void => {
    if (args.text === 'Renewable') {
      args.shape = 'Circle';
    } else if (args.text === 'Non-Renewable') {
      args.shape = 'Triangle';
    }
  };
  const primaryxAxis: AxisModel = { valueType: 'Double' };
  const primaryyAxis: AxisModel = { valueType: 'Double' };

  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='FB Penetration of Internet Audience'
    legendRender={legendRender}>
    <Inject services={[StepAreaSeries, Legend, Tooltip, DataLabel,
  Category]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} type='StepArea' xName='x'
  yName='y' width={2} name='Renewable'>
    </SeriesDirective>
      <SeriesDirective dataSource={data1} type='StepArea' xName='x'
  yName='y' width={2} name='Non-Renewable' >
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
  };
  export default App;
  ReactDOM.render(<App />, document.getElementById("charts"));

```

Tool tip table in React Chart component

You can show the tooltip as table by using template property in tooltip.

Follow the given steps to show the table tooltip,

Step 1:

Initialize the tooltip template div as shown in the following html page,

```
<div id='templateWrap'>
```

Female

<code>\$(x):</code>	<code>\$(y)</code>
---------------------	--------------------

`</div>`

,

Step 2:

To show that tooltip template, set the element id to the `template` property in tooltip.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
StripLine, Legend, Category, Tooltip, DataLabel, LineSeries } from
'@syncfusion/ej2-react-charts';
function App() {
    const data = [
        { x: 1, y: 20 }, { x: 2, y: 22 }, { x: 3, y: 10 }, { x: 4, y: 12 },
        { x: 5, y: 5 },
        { x: 6, y: 15 }, { x: 7, y: 6 }, { x: 8, y: 12 }, { x: 9, y: 34 }, {
x: 10, y: 7 },
    ];
    const primaryxAxis = { title: 'Overs' };
    const primaryyAxis = { title: 'Runs' };
    const template = chartTemplate;
    const tooltip = {
        enable: true,
        template: template
    };
    function chartTemplate(args) {
        return (<div id="templateWrap">
            <table style={{ width: '100%', margin: '5px', border: '1px solid
black', backgroundColor: '#00FFFF' }}>
                <tbody><tr><th colSpan={2}>Female</th></tr>
                <tr><td>{args.x}</td><td>:</td><td>{args.y}</td></tr>
            </tbody>
            </table>
        </div>);
    }
    const marker = { visible: true };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='India Vs Australia 1st match'
tooltip={tooltip}>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category,
StripLine]}/>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' type='Line'
marker={marker}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>;
}
;
```



```
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, StripLine, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection,
StripLinesDirective, StripLineDirective, AxisModel, TooltipSettingsModel}
from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { x: 1, y: 20 }, { x: 2, y: 22 }, { x: 3, y: 10 }, { x: 4, y: 12 }, { x:
5, y: 5 },
    { x: 6, y: 15 }, { x: 7, y: 6 }, { x: 8, y: 12 }, { x: 9, y: 34 }, { x:
10, y: 7 },
  ];
  const primaryxAxis: AxisModel = { title: 'Overs' };
  const primaryyAxis: AxisModel = { title: 'Runs' };
  const template: any = chartTemplate;
  const tooltip: TooltipSettingsModel = {
    enable: true,
    template: template
  };
  function chartTemplate(args: any) {
    return (<div id="templateWrap">
      <table style={{width: '100%', margin: '5px', border: '1px solid black'
, backgroundColor: '#00FFFF'}}>
        <tbody><tr><th colSpan={2}>Female</th></tr>
        <tr><td>{args.x}</td><td>:</td><td>{args.y}</td></tr>
        </tbody>
      </table>
    </div>);
  }
  const marker = { visible: true };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='India Vs Australia 1st match'
    tooltip={tooltip}>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category,
StripLine]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' type='Line'
marker={marker}>
      </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
);
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

Footer in React Chart component

By using `annotation`, you can place any html elements to chart in a desired view.

To create footer and watermark for chart, follow the given steps:

Step 1:

Initialize the custom elements by using the `annotation` property.

By using the `content` option of the annotation object, you can specify the id of the element that needs to be displayed in the chart area as follow,

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ChartAnnotation, AnnotationsDirective, AnnotationDirective, Legend,
Category, Tooltip, DataLabel, SplineSeries } from '@syncfusion/ej2-react-
charts';
function App() {
  const data = [
    { x: 'Sun', y: 15 }, { x: 'Mon', y: 5 },
    { x: 'Tue', y: 32 }, { x: 'Wed', y: 15 },
    { x: 'Thu', y: 29 }, { x: 'Fri', y: 24 },
    { x: 'Sat', y: 18 }
  ];
  const primaryxAxis = { valueType: 'Category' };
  const primaryyAxis = { minimum: 0, maximum: 60, interval: 20 };
  const marker = { visible: true };
  const animation = { enable: true, duration: 1200, delay: 100 };
  const content = chartTemplate;
  function chartTemplate() {
    return (<div className='template'>
      <a href="https://www.syncfusion.com"
target="_blank">www.syncfusion.com</a>
    </div>);
  }
  return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='Olympic Medals'>
    <Inject services={[SplineSeries, Legend, Tooltip, DataLabel, Category,
ChartAnnotation]}/>
    <AnnotationsDirective>
      <AnnotationDirective content={content} coordinateUnits='Point'
x='Wed' y={20}>
    </AnnotationDirective>
      <AnnotationDirective content={content} coordinateUnits='Pixel'
x={440} y={600}>
    </AnnotationDirective>
    </AnnotationsDirective>
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' name='Max
Temp' marker={marker} animation={animation} type='Spline'>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>;
```

```

}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ChartAnnotation, AnnotationsDirective, AnnotationDirective, Legend,
Category, Tooltip, DataLabel, SplineSeries, AxisModel } from '@syncfusion/ej2-
react-charts';
function App() {
    const data: any[] = [
        { x: 'Sun', y: 15 }, { x: 'Mon', y: 5 },
        { x: 'Tue', y: 32 }, { x: 'Wed', y: 15 },
        { x: 'Thu', y: 29 }, { x: 'Fri', y: 24 },
        { x: 'Sat', y: 18 }
    ];
    const primaryxAxis: AxisModel = { valueType: 'Category' };
    const primaryyAxis: AxisModel = { minimum: 0, maximum: 60, interval: 20 };
    const marker = { visible: true };
    const animation = { enable: true, duration: 1200, delay: 100 };
    const content: any = chartTemplate;
    function chartTemplate(): any {
        return (<div className='template'>
            <a href="https://www.syncfusion.com"
target="_blank">www.syncfusion.com</a>
            </div>);
    }
    return <ChartComponent id='charts'
        primaryXAxis={primaryxAxis}
        primaryYAxis={primaryyAxis}
        title='Olympic Medals'>
        <Inject services={[SplineSeries, Legend, Tooltip, DataLabel, Category,
ChartAnnotation]} />
        <AnnotationsDirective>
            <AnnotationDirective content={content} coordinateUnits='Point'
x='Wed' y={20}>
            </AnnotationDirective>
            <AnnotationDirective content={content} coordinateUnits='Pixel'
x={440} y={600}>
            </AnnotationDirective>
        </AnnotationsDirective>
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' name='Max
Temp' marker={marker}
            animation={animation} type='Spline'>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
    </return>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Threshold in React Chart component

You can mark a threshold in chart by using the `stripline`.

To mark a threshold in chart, follow the given steps:

Step 1:

By using the start and end properties of `striplines` object in vertical axis, you can mark the threshold for y values of the series.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
StripLine, Legend, Category, Tooltip, DataLabel, LineSeries } from
'syncfusion/ej2-react-charts';
function App() {
    const data = [
        { x: 1, y: 20 }, { x: 2, y: 22 }, { x: 3, y: 0 }, { x: 4, y: 12 }, {
x: 5, y: 5 },
        { x: 6, y: 15 }, { x: 7, y: 6 }, { x: 8, y: 12 }, { x: 9, y: 34 }, {
x: 10, y: 7 },
    ];
    const primaryxAxis = { title: 'Overs' };
    const primaryyAxis = { title: 'Runs', striplines: [{ start: 15, end:
15.1, color: '#ff512f', visible: true }] };
    const marker = { visible: true };
    return <ChartComponent id='charts' primaryXAxis={primaryxAxis}
primaryYAxis={primaryyAxis} title='India Vs Australia 1st match' width='650'
height='350'>
        <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category,
StripLine]} />
        <SeriesCollectionDirective>
            <SeriesDirective dataSource={data} xName='x' yName='y' type='Line'
Marker={marker}>
                </SeriesDirective>
            </SeriesCollectionDirective>
        </ChartComponent>;
    }
    ;
    export default App;
    ReactDOM.render(<App />, document.getElementById("charts"));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, StripLine, AxisModel, ColumnSeries,
Legend, Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries,
Selection, StripLinesDirective, StripLineDirective } from 'syncfusion/ej2-
react-charts';
function App() {
    const data: any[] = [
        { x: 1, y: 20 }, { x: 2, y: 22 }, { x: 3, y: 0 }, { x: 4, y: 12 }, { x:
5, y: 5 },
```

```

    { x: 6, y: 15 }, { x: 7, y: 6 }, { x: 8, y: 12 }, { x: 9, y: 34 }, { x:
10, y: 7 },
  ];
  const primaryxAxis: AxisModel = { title: 'Overs' };
  const primaryyAxis: AxisModel = { title: 'Runs', stripLines: [{ start: 15,
end: 15.1, color: '#ff512f', visible: true }] };
  const marker = { visible: true };
  return <ChartComponent id='charts'
    primaryXAxis={primaryxAxis}
    primaryYAxis={primaryyAxis}
    title='India Vs Australia 1st match' width='650' height='350'>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category,
StripLine]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' type='Line'
Marker={marker}>
    </SeriesDirective>
    </SeriesCollectionDirective>
  </ChartComponent>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));

```

Grid data chart in React Chart component

You can visualize the data that returned by grid in chart.

To visualize the data in chart, follow the given steps:

Step 1:

Initialize the grid with datasource.

Step 2:

By using the grid's `actionComplete` event and `getCurrentViewRecords` method, you can get the current page records.

By using the grid's `databound` event, you can update the current page records into the chart's datasource and visualize the grid data in chart.

DATASOURCE.JSX

```

export let productData = [
  {
    'ProductID': 1,
    'ProductName': 'Chai',
    'SupplierID': 1,
    'QuantityPerUnit': '10 boxes x 20 bags',
    'UnitPrice': 18.00,
    'UnitsInStock': 39,
    'Discontinued': true
  },
  {
    'ProductID': 2,
    'ProductName': 'Chang',
    'SupplierID': 1,
    'QuantityPerUnit': '24 - 12 oz bottles',

```

```

        'UnitPrice': 19.00,
        'UnitsInStock': 17,
        'Discontinued': true
    },
    {
        'ProductID': 3,
        'ProductName': 'Aniseed Syrup',
        'SupplierID': 1,
        'QuantityPerUnit': '12 - 550 ml bottles',
        'UnitPrice': 10.00,
        'UnitsInStock': 13,
        'Discontinued': true
    },
    {
        'ProductID': 4,
        'ProductName': 'Chef Anton\'s Cajun Seasoning',
        'SupplierID': 2,
        'QuantityPerUnit': '48 - 6 oz jars',
        'UnitPrice': 22.00,
        'UnitsInStock': 53,
        'Discontinued': true
    },
    {
        'ProductID': 5,
        'ProductName': 'Chef Anton\'s Gumbo Mix',
        'SupplierID': 2,
        'QuantityPerUnit': '36 boxes',
        'UnitPrice': 21.35,
        'UnitsInStock': 0,
        'Discontinued': true
    },
    {
        'ProductID': 6,
        'ProductName': 'Grandma\'s Boysenberry Spread',
        'SupplierID': 3,
        'QuantityPerUnit': '12 - 8 oz jars',
        'UnitPrice': 25.00,
        'UnitsInStock': 120,
        'Discontinued': false
    },
    {
        'ProductID': 7,
        'ProductName': 'Uncle Bob\'s Organic Dried Pears',
        'SupplierID': 3,
        'QuantityPerUnit': '12 - 1 lb pkgs.',
        'UnitPrice': 30.00,
        'UnitsInStock': 15,
        'Discontinued': false
    },
    {
        'ProductID': 8,
        'ProductName': 'Northwoods Cranberry Sauce',
        'SupplierID': 3,
        'QuantityPerUnit': '12 - 12 oz jars',
        'UnitPrice': 40.00,
        'UnitsInStock': 6,
        'Discontinued': false
    }

```

```

    },
    {
        'ProductID': 9,
        'ProductName': 'Mishi Kobe Niku',
        'SupplierID': 4,
        'QuantityPerUnit': '18 - 500 g pkgs.',
        'UnitPrice': 97.00,
        'UnitsInStock': 29,
        'Discontinued': true
    },
    {
        'ProductID': 10,
        'ProductName': 'Ikura',
        'SupplierID': 4,
        'QuantityPerUnit': '12 - 200 ml jars',
        'UnitPrice': 31.00,
        'UnitsInStock': 31,
        'Discontinued': false
    },
    {
        'ProductID': 11,
        'ProductName': 'Queso Cabrales',
        'SupplierID': 5,
        'QuantityPerUnit': '1 kg pkg.',
        'UnitPrice': 21.00,
        'UnitsInStock': 22,
        'Discontinued': false
    },
    {
        'ProductID': 12,
        'ProductName': 'Queso Manchego La Pastora',
        'SupplierID': 5,
        'QuantityPerUnit': '10 - 500 g pkgs.',
        'UnitPrice': 38.00,
        'UnitsInStock': 86,
        'Discontinued': false
    },
    {
        'ProductID': 13,
        'ProductName': 'Konbu',
        'SupplierID': 6,
        'QuantityPerUnit': '2 kg box',
        'UnitPrice': 6.00,
        'UnitsInStock': 24,
        'Discontinued': true
    },
    {
        'ProductID': 14,
        'ProductName': 'Tofu',
        'SupplierID': 6,
        'QuantityPerUnit': '40 - 100 g pkgs.',
        'UnitPrice': 23.25,
        'UnitsInStock': 35,
        'Discontinued': true
    },
    {
        'ProductID': 15,

```

```

        'ProductName': 'Genen Shouyu',
        'SupplierID': 6,
        'QuantityPerUnit': '24 - 250 ml bottles',
        'UnitPrice': 15.50,
        'UnitsInStock': 39,
        'Discontinued': true
    },
    {
        'ProductID': 16,
        'ProductName': 'Pavlova',
        'SupplierID': 7,
        'QuantityPerUnit': '32 - 500 g boxes',
        'UnitPrice': 17.45,
        'UnitsInStock': 29,
        'Discontinued': true
    },
    {
        'ProductID': 17,
        'ProductName': 'Alice Mutton',
        'SupplierID': 7,
        'QuantityPerUnit': '20 - 1 kg tins',
        'UnitPrice': 39.00,
        'UnitsInStock': 0,
        'Discontinued': true
    },
    {
        'ProductID': 18,
        'ProductName': 'Carnarvon Tigers',
        'SupplierID': 7,
        'QuantityPerUnit': '16 kg pkg.',
        'UnitPrice': 62.50,
        'UnitsInStock': 42,
        'Discontinued': false
    },
    {
        'ProductID': 19,
        'ProductName': 'Teatime Chocolate Biscuits',
        'SupplierID': 8,
        'QuantityPerUnit': '10 boxes x 12 pieces',
        'UnitPrice': 9.20,
        'UnitsInStock': 25,
        'Discontinued': false
    },
    {
        'ProductID': 20,
        'ProductName': 'Sir Rodney\'s Marmalade',
        'SupplierID': 8,
        'QuantityPerUnit': '30 gift boxes',
        'UnitPrice': 81.00,
        'UnitsInStock': 40,
        'Discontinued': false
    },
    {
        'ProductID': 21,
        'ProductName': 'Sir Rodney\'s Scones',
        'SupplierID': 8,
        'QuantityPerUnit': '24 pkgs. x 4 pieces',
    }

```



```
'UnitPrice': 10.00,  
'UnitsInStock': 3,  
'Discontinued': false  
},  
{  
  'ProductID': 22,  
  'ProductName': 'Gustaf\'s Knäckebröd',  
  'SupplierID': 9,  
  'QuantityPerUnit': '24 - 500 g pkgs.',  
  'UnitPrice': 21.00,  
  'UnitsInStock': 104,  
  'Discontinued': false  
},  
{  
  'ProductID': 23,  
  'ProductName': 'Tunnbröd',  
  'SupplierID': 9,  
  'QuantityPerUnit': '12 - 250 g pkgs.',  
  'UnitPrice': 9.00,  
  'UnitsInStock': 61,  
  'Discontinued': false  
},  
{  
  'ProductID': 24,  
  'ProductName': 'Guaraná Fantástica',  
  'SupplierID': 10,  
  'QuantityPerUnit': '12 - 355 ml cans',  
  'UnitPrice': 4.50,  
  'UnitsInStock': 20,  
  'Discontinued': true  
},  
{  
  'ProductID': 25,  
  'ProductName': 'NuNuCa Nuß-Nougat-Creme',  
  'SupplierID': 11,  
  'QuantityPerUnit': '20 - 450 g glasses',  
  'UnitPrice': 14.00,  
  'UnitsInStock': 76,  
  'Discontinued': false  
},  
{  
  'ProductID': 26,  
  'ProductName': 'Gumbär Gummibärchen',  
  'SupplierID': 11,  
  'QuantityPerUnit': '100 - 250 g bags',  
  'UnitPrice': 31.23,  
  'UnitsInStock': 15,  
  'Discontinued': true  
},  
{  
  'ProductID': 27,  
  'ProductName': 'Schoggi Schokolade',  
  'SupplierID': 11,  
  'QuantityPerUnit': '100 - 100 g pieces',  
  'UnitPrice': 43.90,  
  'UnitsInStock': 49,  
  'Discontinued': true
```

```

    },
    {
        'ProductID': 28,
        'ProductName': 'Rössle Sauerkraut',
        'SupplierID': 12,
        'QuantityPerUnit': '25 - 825 g cans',
        'UnitPrice': 45.60,
        'UnitsInStock': 26,
        'Discontinued': true
    },
    {
        'ProductID': 29,
        'ProductName': 'Thüringer Rostbratwurst',
        'SupplierID': 12,
        'QuantityPerUnit': '50 bags x 30 sausgs.',
        'UnitPrice': 123.79,
        'UnitsInStock': 0,
        'Discontinued': true
    },
    {
        'ProductID': 30,
        'ProductName': 'Nord-Ost Matjeshering',
        'SupplierID': 13,
        'QuantityPerUnit': '10 - 200 g glasses',
        'UnitPrice': 25.89,
        'UnitsInStock': 10,
        'Discontinued': true
    },
    {
        'ProductID': 31,
        'ProductName': 'Gorgonzola Telino',
        'SupplierID': 14,
        'QuantityPerUnit': '12 - 100 g pkgs',
        'UnitPrice': 12.50,
        'UnitsInStock': 0,
        'Discontinued': true
    },
    {
        'ProductID': 32,
        'ProductName': 'Mascarpone Fabioli',
        'SupplierID': 14,
        'QuantityPerUnit': '24 - 200 g pkgs.',
        'UnitPrice': 32.00,
        'UnitsInStock': 9,
        'Discontinued': false
    },
    {
        'ProductID': 33,
        'ProductName': 'Geitost',
        'SupplierID': 15,
        'QuantityPerUnit': '500 g',
        'UnitPrice': 2.50,
        'UnitsInStock': 112,
        'Discontinued': false
    },
    {
        'ProductID': 34,

```

```

    'ProductName': 'Sasquatch Ale',
    'SupplierID': 16,
    'QuantityPerUnit': '24 - 12 oz bottles',
    'UnitPrice': 14.00,
    'UnitsInStock': 111,
    'Discontinued': false
  },
  {
    'ProductID': 35,
    'ProductName': 'Steeleye Stout',
    'SupplierID': 16,
    'QuantityPerUnit': '24 - 12 oz bottles',
    'UnitPrice': 18.00,
    'UnitsInStock': 20,
    'Discontinued': false
  },
  {
    'ProductID': 36,
    'ProductName': 'Inlagd Sill',
    'SupplierID': 17,
    'QuantityPerUnit': '24 - 250 g jars',
    'UnitPrice': 19.00,
    'UnitsInStock': 112,
    'Discontinued': false
  },
  {
    'ProductID': 37,
    'ProductName': 'Gravad lax',
    'SupplierID': 17,
    'QuantityPerUnit': '12 - 500 g pkgs.',
    'UnitPrice': 26.00,
    'UnitsInStock': 11,
    'Discontinued': false
  },
  {
    'ProductID': 38,
    'ProductName': 'Côte de Blaye',
    'SupplierID': 18,
    'QuantityPerUnit': '12 - 75 cl bottles',
    'UnitPrice': 263.50,
    'UnitsInStock': 17,
    'Discontinued': false
  },
  {
    'ProductID': 39,
    'ProductName': 'Chartreuse verte',
    'SupplierID': 18,
    'QuantityPerUnit': '750 cc per bottle',
    'UnitPrice': 18.00,
    'UnitsInStock': 69,
    'Discontinued': true
  },
  {
    'ProductID': 40,
    'ProductName': 'Boston Crab Meat',
    'SupplierID': 19,
    'QuantityPerUnit': '24 - 4 oz tins',

```

```

      'UnitPrice': 18.40,
      'UnitsInStock': 123,
      'Discontinued': true
    },
    {
      'ProductID': 41,
      'ProductName': 'Jack\'s New England Clam Chowder',
      'SupplierID': 19,
      'QuantityPerUnit': '12 - 12 oz cans',
      'UnitPrice': 9.65,
      'UnitsInStock': 85,
      'Discontinued': false
    },
    {
      'ProductID': 42,
      'ProductName': 'Singaporean Hokkien Fried Mee',
      'SupplierID': 20,
      'QuantityPerUnit': '32 - 1 kg pkgs.',
      'UnitPrice': 14.00,
      'UnitsInStock': 26,
      'Discontinued': true
    },
    {
      'ProductID': 43,
      'ProductName': 'Ipoh Coffee',
      'SupplierID': 20,
      'QuantityPerUnit': '16 - 500 g tins',
      'UnitPrice': 46.00,
      'UnitsInStock': 17,
      'Discontinued': false
    },
    {
      'ProductID': 44,
      'ProductName': 'Gula Malacca',
      'SupplierID': 20,
      'QuantityPerUnit': '20 - 2 kg bags',
      'UnitPrice': 19.45,
      'UnitsInStock': 27,
      'Discontinued': false
    },
    {
      'ProductID': 45,
      'ProductName': 'Rogede sild',
      'SupplierID': 21,
      'QuantityPerUnit': '1k pkg.',
      'UnitPrice': 9.50,
      'UnitsInStock': 5,
      'Discontinued': true
    },
    {
      'ProductID': 46,
      'ProductName': 'Spegesild',
      'SupplierID': 21,
      'QuantityPerUnit': '4 - 450 g glasses',
      'UnitPrice': 12.00,
      'UnitsInStock': 95,
      'Discontinued': true
    }
  ]

```

```

    },
    {
        'ProductID': 47,
        'ProductName': 'Zaanse koeken',
        'SupplierID': 22,
        'QuantityPerUnit': '10 - 4 oz boxes',
        'UnitPrice': 9.50,
        'UnitsInStock': 36,
        'Discontinued': true
    },
    {
        'ProductID': 48,
        'ProductName': 'Chocolade',
        'SupplierID': 22,
        'QuantityPerUnit': '10 pkgs.',
        'UnitPrice': 12.75,
        'UnitsInStock': 15,
        'Discontinued': true
    },
    {
        'ProductID': 49,
        'ProductName': 'Maxilaku',
        'SupplierID': 23,
        'QuantityPerUnit': '24 - 50 g pkgs.',
        'UnitPrice': 20.00,
        'UnitsInStock': 10,
        'Discontinued': false
    },
    {
        'ProductID': 50,
        'ProductName': 'Valkoinen suklaa',
        'SupplierID': 23,
        'QuantityPerUnit': '12 - 100 g bars',
        'UnitPrice': 16.25,
        'UnitsInStock': 65,
        'Discontinued': false
    },
    {
        'ProductID': 51,
        'ProductName': 'Manjimup Dried Apples',
        'SupplierID': 24,
        'QuantityPerUnit': '50 - 300 g pkgs.',
        'UnitPrice': 53.00,
        'UnitsInStock': 20,
        'Discontinued': false
    },
    {
        'ProductID': 52,
        'ProductName': 'Filo Mix',
        'SupplierID': 24,
        'QuantityPerUnit': '16 - 2 kg boxes',
        'UnitPrice': 7.00,
        'UnitsInStock': 38,
        'Discontinued': true
    },
    {
        'ProductID': 53,

```

```

        'ProductName': 'Perth Pasties',
        'SupplierID': 24,
        'QuantityPerUnit': '48 pieces',
        'UnitPrice': 32.80,
        'UnitsInStock': 0,
        'Discontinued': true
    },
    {
        'ProductID': 54,
        'ProductName': 'Tourtière',
        'SupplierID': 25,
        'QuantityPerUnit': '16 pies',
        'UnitPrice': 7.45,
        'UnitsInStock': 21,
        'Discontinued': true
    },
    {
        'ProductID': 55,
        'ProductName': 'Pâté chinois',
        'SupplierID': 25,
        'QuantityPerUnit': '24 boxes x 2 pies',
        'UnitPrice': 24.00,
        'UnitsInStock': 115,
        'Discontinued': true
    },
    {
        'ProductID': 56,
        'ProductName': 'Gnocchi di nonna Alice',
        'SupplierID': 26,
        'QuantityPerUnit': '24 - 250 g pkgs.',
        'UnitPrice': 38.00,
        'UnitsInStock': 21,
        'Discontinued': false
    },
    {
        'ProductID': 57,
        'ProductName': 'Ravioli Angelo',
        'SupplierID': 26,
        'QuantityPerUnit': '24 - 250 g pkgs.',
        'UnitPrice': 19.50,
        'UnitsInStock': 36,
        'Discontinued': false
    },
    {
        'ProductID': 58,
        'ProductName': 'Escargots de Bourgogne',
        'SupplierID': 27,
        'QuantityPerUnit': '24 pieces',
        'UnitPrice': 13.25,
        'UnitsInStock': 62,
        'Discontinued': false
    },
    {
        'ProductID': 59,
        'ProductName': 'Raclette Courdavault',
        'SupplierID': 28,
        'QuantityPerUnit': '5 kg pkg.',

```

```
'UnitPrice': 55.00,  
'UnitsInStock': 79,  
'Discontinued': false  
},  
{  
  'ProductID': 60,  
  'ProductName': 'Camembert Pierrot',  
  'SupplierID': 28,  
  'QuantityPerUnit': '15 - 300 g rounds',  
  'UnitPrice': 34.00,  
  'UnitsInStock': 19,  
  'Discontinued': false  
},  
{  
  'ProductID': 61,  
  'ProductName': 'Sirop d\'érable',  
  'SupplierID': 29,  
  'QuantityPerUnit': '24 - 500 ml bottles',  
  'UnitPrice': 28.50,  
  'UnitsInStock': 113,  
  'Discontinued': false  
},  
{  
  'ProductID': 62,  
  'ProductName': 'Tarte au sucre',  
  'SupplierID': 29,  
  'QuantityPerUnit': '48 pies',  
  'UnitPrice': 49.30,  
  'UnitsInStock': 17,  
  'Discontinued': false  
},  
{  
  'ProductID': 63,  
  'ProductName': 'Veggie-spread',  
  'SupplierID': 7,  
  'QuantityPerUnit': '15 - 625 g jars',  
  'UnitPrice': 43.90,  
  'UnitsInStock': 24,  
  'Discontinued': true  
},  
{  
  'ProductID': 64,  
  'ProductName': 'Wimmers gute Semmelknödel',  
  'SupplierID': 12,  
  'QuantityPerUnit': '20 bags x 4 pieces',  
  'UnitPrice': 33.25,  
  'UnitsInStock': 22,  
  'Discontinued': true  
},  
{  
  'ProductID': 65,  
  'ProductName': 'Louisiana Fiery Hot Pepper Sauce',  
  'SupplierID': 2,  
  'QuantityPerUnit': '32 - 8 oz bottles',  
  'UnitPrice': 21.05,  
  'UnitsInStock': 76,  
  'Discontinued': true
```

```

    },
    {
        'ProductID': 66,
        'ProductName': 'Louisiana Hot Spiced Okra',
        'SupplierID': 2,
        'QuantityPerUnit': '24 - 8 oz jars',
        'UnitPrice': 17.00,
        'UnitsInStock': 4,
        'Discontinued': false
    },
    {
        'ProductID': 67,
        'ProductName': 'Laughing Lumberjack Lager',
        'SupplierID': 16,
        'QuantityPerUnit': '24 - 12 oz bottles',
        'UnitPrice': 14.00,
        'UnitsInStock': 52,
        'Discontinued': false
    },
    {
        'ProductID': 68,
        'ProductName': 'Scottish Longbreads',
        'SupplierID': 8,
        'QuantityPerUnit': '10 boxes x 8 pieces',
        'UnitPrice': 12.50,
        'UnitsInStock': 6,
        'Discontinued': false
    },
    {
        'ProductID': 69,
        'ProductName': 'Gudbrandsdalsost',
        'SupplierID': 15,
        'QuantityPerUnit': '10 kg pkg.',
        'UnitPrice': 36.00,
        'UnitsInStock': 26,
        'Discontinued': false
    },
    {
        'ProductID': 70,
        'ProductName': 'Outback Lager',
        'SupplierID': 7,
        'QuantityPerUnit': '24 - 355 ml bottles',
        'UnitPrice': 15.00,
        'UnitsInStock': 15,
        'Discontinued': false
    },
    {
        'ProductID': 71,
        'ProductName': 'Flotemysost',
        'SupplierID': 15,
        'QuantityPerUnit': '10 - 500 g pkgs.',
        'UnitPrice': 21.50,
        'UnitsInStock': 26,
        'Discontinued': true
    },
    {
        'ProductID': 72,

```



```

        'ProductName': 'Mozzarella di Giovanni',
        'SupplierID': 14,
        'QuantityPerUnit': '24 - 200 g pkgs.',
        'UnitPrice': 34.80,
        'UnitsInStock': 14,
        'Discontinued': true
    },
    {
        'ProductID': 73,
        'ProductName': 'Röd Kaviar',
        'SupplierID': 17,
        'QuantityPerUnit': '24 - 150 g jars',
        'UnitPrice': 15.00,
        'UnitsInStock': 101,
        'Discontinued': true
    },
    {
        'ProductID': 74,
        'ProductName': 'Longlife Tofu',
        'SupplierID': 4,
        'QuantityPerUnit': '5 kg pkg.',
        'UnitPrice': 10.00,
        'UnitsInStock': 4,
        'Discontinued': true
    },
    {
        'ProductID': 75,
        'ProductName': 'Rhönbräu Klosterbier',
        'SupplierID': 12,
        'QuantityPerUnit': '24 - 0.5 l bottles',
        'UnitPrice': 7.75,
        'UnitsInStock': 125,
        'Discontinued': true
    },
    {
        'ProductID': 76,
        'ProductName': 'Lakkalikööri',
        'SupplierID': 23,
        'QuantityPerUnit': '500 ml',
        'UnitPrice': 18.00,
        'UnitsInStock': 57,
        'Discontinued': true
    },
    {
        'ProductID': 77,
        'ProductName': 'Original Frankfurter grüne Soße',
        'SupplierID': 12,
        'QuantityPerUnit': '12 boxes',
        'UnitPrice': 13.00,
        'UnitsInStock': 32,
        'Discontinued': false
    }
];
export let employeeData = [{
    'EmployeeID': 1,
    'LastName': 'Davolio',
    'FirstName': 'Nancy',

```

```

        'Title': 'Sales Representative',
        'TitleOfCourtesy': 'Ms.',
        'BirthDate': new Date(-664743600000),
        'HireDate': new Date(704692800000),
        'Address': '507 - 20th Ave. E.\r\nApt. 2A',
        'City': 'Seattle',
        'Region': 'WA',
        'PostalCode': '98122',
        'Country': 'USA',
        'HomePhone': '(206) 555-9857',
        'Extension': '5467',
        'Photo': { 'Length': 21626 },
        'Notes': 'Education includes a BA in psychology from Colorado State
University in 1970. She also completed\
\'The Art of the Cold Call.\' Nancy is a member of Toastmasters
International.',
        'ReportsTo': 2,
        'PhotoPath': 'http://accweb/emmployees/davolio.bmp'
    },
    {
        'EmployeeID': 2,
        'LastName': 'Fuller',
        'FirstName': 'Andrew',
        'Title': 'Vice President, Sales',
        'TitleOfCourtesy': 'Dr.',
        'BirthDate': new Date(-563828400000),
        'HireDate': new Date(713764800000),
        'Address': '908 W. Capital Way',
        'City': 'Tacoma',
        'Region': 'WA',
        'PostalCode': '98401',
        'Country': 'USA',
        'HomePhone': '(206) 555-9482',
        'Extension': '3457',
        'Photo': { 'Length': 21626 },
        'Notes': 'Andrew received his BTS commercial in 1974 and a Ph.D. in
international marketing from the University of \
Dallas in 1981. He is fluent in French and Italian and reads German.
He joined the company as a sales representative, \
was promoted to sales manager in January 1992 and to vice president of
sales in March 1993. Andrew is a member of the \
Sales Management Roundtable, the Seattle Chamber of Commerce, and the
Pacific Rim Importers Association.',
        'ReportsTo': 0,
        'PhotoPath': 'http://accweb/emmployees/fuller.bmp'
    },
    {
        'EmployeeID': 3,
        'LastName': 'Leverling',
        'FirstName': 'Janet',
        'Title': 'Sales Representative',
        'TitleOfCourtesy': 'Ms.',
        'BirthDate': new Date(-200088000000),
        'HireDate': new Date(702104400000),
        'Address': '722 Moss Bay Blvd.',
        'City': 'Kirkland',
        'Region': 'WA',

```

```

        'PostalCode': '98033',
        'Country': 'USA',
        'HomePhone': '(206) 555-3412',
        'Extension': '3355',
        'Photo': { 'Length': 21722 },
        'Notes': 'Janet has a BS degree in chemistry from Boston College
(1984). \
        She has also completed a certificate program in food retailing
management.\
        Janet was hired as a sales associate in 1991 and promoted to sales
representative in February 1992.',
        'ReportsTo': 2,
        'PhotoPath': 'http://accweb/emmployees/leverling.bmp'
    },
    {
        'EmployeeID': 4,
        'LastName': 'Peacock',
        'FirstName': 'Margaret',
        'Title': 'Sales Representative',
        'TitleOfCourtesy': 'Mrs.',
        'BirthDate': new Date(-1018814400000),
        'HireDate': new Date(736401600000),
        'Address': '4110 Old Redmond Rd.',
        'City': 'Redmond',
        'Region': 'WA',
        'PostalCode': '98052',
        'Country': 'USA',
        'HomePhone': '(206) 555-8122',
        'Extension': '5176',
        'Photo': { 'Length': 21626 },
        'Notes': 'Margaret holds a BA in English literature from Concordia
College (1958) and an MA from the American \
        Institute of Culinary Arts (1966). She was assigned to the London
office temporarily from July through November 1992.',
        'ReportsTo': 2,
        'PhotoPath': 'http://accweb/emmployees/peacock.bmp'
    },
    {
        'EmployeeID': 5,
        'LastName': 'Buchanan',
        'FirstName': 'Steven',
        'Title': 'Sales Manager',
        'TitleOfCourtesy': 'Mr.',
        'BirthDate': new Date(-468010800000),
        'HireDate': new Date(750830400000),
        'Address': '14 Garrett Hill',
        'City': 'London',
        'Region': null,
        'PostalCode': 'SW1 8JR',
        'Country': 'UK',
        'HomePhone': '(71) 555-4848',
        'Extension': '3453',
        'Photo': { 'Length': 21626 },
        'Notes': 'Steven Buchanan graduated from St. Andrews University,
Scotland, with a BSC degree in 1976. Upon joining the company as \
        a sales representative in 1992, he spent 6 months in an orientation
program at the Seattle office and then returned to his permanent \

```

```

    post in London. He was promoted to sales manager in March 1993. Mr.
    Buchanan has completed the courses \'Successful \
    Telemarketing\' and \'International Sales Management.\' He is fluent in
    French.',
    'ReportsTo': 2,
    'PhotoPath': 'http://accweb/emmployees/buchanan.bmp'
  },
  {
    'EmployeeID': 6,
    'LastName': 'Suyama',
    'FirstName': 'Michael',
    'Title': 'Sales Representative',
    'TitleOfCourtesy': 'Mr.',
    'BirthDate': new Date(-205185600000),
    'HireDate': new Date(750830400000),
    'Address': 'Coventry House\r\nMiner Rd.',
    'City': 'London',
    'Region': null,
    'PostalCode': 'EC2 7JR',
    'Country': 'UK',
    'HomePhone': '(71) 555-7773',
    'Extension': '428',
    'Photo': { 'Length': 21626 },
    'Notes': 'Michael is a graduate of Sussex University (MA, economics,
    1983) and the University of California at Los Angeles \
    (MBA, marketing, 1986). He has also taken the courses \'Multi-Cultural
    Selling\' and \'Time Management for the Sales Professional.\' \
    He is fluent in Japanese and can read and write French, Portuguese, and
    Spanish.',
    'ReportsTo': 5,
    'PhotoPath': 'http://accweb/emmployees/davolio.bmp'
  },
  {
    'EmployeeID': 7,
    'LastName': 'King',
    'FirstName': 'Robert',
    'Title': 'Sales Representative',
    'TitleOfCourtesy': 'Mr.',
    'BirthDate': new Date(-302731200000),
    'HireDate': new Date(757486800000),
    'Address': 'Edgeham Hollow\r\nWinchester Way',
    'City': 'London',
    'Region': null,
    'PostalCode': 'RG1 9SP',
    'Country': 'UK',
    'HomePhone': '(71) 555-5598',
    'Extension': '465',
    'Photo': { 'Length': 21626 },
    'Notes': 'Robert King served in the Peace Corps and traveled
    extensively before completing his degree in English at the \
    University of Michigan in 1992, the year he joined the company. After
    completing a course entitled \'Selling in Europe,\' \
    he was transferred to the London office in March 1993.',
    'ReportsTo': 5,
    'PhotoPath': 'http://accweb/emmployees/davolio.bmp'
  },
  {

```

```

        'EmployeeID': 8,
        'LastName': 'Callahan',
        'FirstName': 'Laura',
        'Title': 'Inside Sales Coordinator',
        'TitleOfCourtesy': 'Ms.',
        'BirthDate': new Date(-377982000000),
        'HireDate': new Date(762843600000),
        'Address': '4726 - 11th Ave. N.E.',
        'City': 'Seattle',
        'Region': 'WA',
        'PostalCode': '98105',
        'Country': 'USA',
        'HomePhone': '(206) 555-1189',
        'Extension': '2344',
        'Photo': { 'Length': 21626 },
        'Notes': 'Laura received a BA in psychology from the University of
Washington. She has also completed a course in business \
French. She reads and writes French.',
        'ReportsTo': 2,
        'PhotoPath': 'http://accweb/emmployees/davolio.bmp'
    },
    {
        'EmployeeID': 9,
        'LastName': 'Dodsworth',
        'FirstName': 'Anne',
        'Title': 'Sales Representative',
        'TitleOfCourtesy': 'Ms.',
        'BirthDate': new Date(-123966000000),
        'HireDate': new Date(784875600000),
        'Address': '7 Houndstooth Rd.',
        'City': 'London',
        'Region': null,
        'PostalCode': 'WG2 7LT',
        'Country': 'UK',
        'HomePhone': '(71) 555-4444',
        'Extension': '452',
        'Photo': { 'Length': 21626 },
        'Notes': 'Anne has a BA degree in English from St. Lawrence College.
She is fluent in French and German.',
        'ReportsTo': 5,
        'PhotoPath': 'http://accweb/emmployees/davolio.bmp'
    }
    ]];
let stringData = JSON.stringify([
    {
        'OrderID': 10248,
        'CustomerID': 'VINET',
        'OrderDate': '1996-07-04T00:00:00.000Z',
        'ShippedDate': '1996-07-16T00:00:00.000Z',
        'Freight': 32.38,
        'ShipName': 'Vins et alcools Chevalier',
        'ShipAddress': '59 rue de l\'Abbaye',
        'ShipCity': 'Reims',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10249,

```

```

      'CustomerID': 'TOMSP',
      'OrderDate': '1996-07-05T00:00:00.000Z',
      'ShippedDate': '1996-07-10T00:00:00.000Z',
      'Freight': 11.61,
      'ShipName': 'Toms Spezialitäten',
      'ShipAddress': 'Luisenstr. 48',
      'ShipCity': 'Münster',
      'ShipRegion': null,
      'ShipCountry': 'Germany'
    },
    {
      'OrderID': 10250,
      'CustomerID': 'HANAR',
      'OrderDate': '1996-07-08T00:00:00.000Z',
      'ShippedDate': '1996-07-12T00:00:00.000Z',
      'Freight': 65.83,
      'ShipName': 'Hanari Carnes',
      'ShipAddress': 'Rua do Paço, 67',
      'ShipCity': 'Rio de Janeiro',
      'ShipRegion': 'RJ',
      'ShipCountry': 'Brazil'
    },
    {
      'OrderID': 10251,
      'CustomerID': 'VICTE',
      'OrderDate': '1996-07-08T00:00:00.000Z',
      'ShippedDate': '1996-07-15T00:00:00.000Z',
      'Freight': 41.34,
      'ShipName': 'Victuailles en stock',
      'ShipAddress': '2, rue du Commerce',
      'ShipCity': 'Lyon',
      'ShipRegion': null,
      'ShipCountry': 'France'
    },
    {
      'OrderID': 10252,
      'CustomerID': 'SUPRD',
      'OrderDate': '1996-07-09T00:00:00.000Z',
      'ShippedDate': '1996-07-11T00:00:00.000Z',
      'Freight': 51.3,
      'ShipName': 'Suprêmes délices',
      'ShipAddress': 'Boulevard Tirou, 255',
      'ShipCity': 'Charleroi',
      'ShipRegion': null,
      'ShipCountry': 'Belgium'
    },
    {
      'OrderID': 10253,
      'CustomerID': 'HANAR',
      'OrderDate': '1996-07-10T00:00:00.000Z',
      'ShippedDate': '1996-07-16T00:00:00.000Z',
      'Freight': 58.17,
      'ShipName': 'Hanari Carnes',
      'ShipAddress': 'Rua do Paço, 67',
      'ShipCity': 'Rio de Janeiro',
      'ShipRegion': 'RJ',
      'ShipCountry': 'Brazil'
    }
  ]

```

```

},
{
  'OrderID': 10254,
  'CustomerID': 'CHOPS',
  'OrderDate': '1996-07-11T00:00:00.000Z',
  'ShippedDate': '1996-07-23T00:00:00.000Z',
  'Freight': 22.98,
  'ShipName': 'Chop-suey Chinese',
  'ShipAddress': 'Hauptstr. 31',
  'ShipCity': 'Bern',
  'ShipRegion': null,
  'ShipCountry': 'Switzerland'
},
{
  'OrderID': 10255,
  'CustomerID': 'RICSU',
  'OrderDate': '1996-07-12T00:00:00.000Z',
  'ShippedDate': '1996-07-15T00:00:00.000Z',
  'Freight': 148.33,
  'ShipName': 'Richter Supermarkt',
  'ShipAddress': 'Starenweg 5',
  'ShipCity': 'Genève',
  'ShipRegion': null,
  'ShipCountry': 'Switzerland'
},
{
  'OrderID': 10256,
  'CustomerID': 'WELLI',
  'OrderDate': '1996-07-15T00:00:00.000Z',
  'ShippedDate': '1996-07-17T00:00:00.000Z',
  'Freight': 13.97,
  'ShipName': 'Wellington Importadora',
  'ShipAddress': 'Rua do Mercado, 12',
  'ShipCity': 'Resende',
  'ShipRegion': 'SP',
  'ShipCountry': 'Brazil'
},
{
  'OrderID': 10257,
  'CustomerID': 'HILAA',
  'OrderDate': '1996-07-16T00:00:00.000Z',
  'ShippedDate': '1996-07-22T00:00:00.000Z',
  'Freight': 81.91,
  'ShipName': 'HILARION-Abastos',
  'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
  'ShipCity': 'San Cristóbal',
  'ShipRegion': 'Táchira',
  'ShipCountry': 'Venezuela'
},
{
  'OrderID': 10258,
  'CustomerID': 'ERNSH',
  'OrderDate': '1996-07-17T00:00:00.000Z',
  'ShippedDate': '1996-07-23T00:00:00.000Z',
  'Freight': 140.51,
  'ShipName': 'Ernst Handel',
  'ShipAddress': 'Kirchgasse 6',

```

```

        'ShipCity': 'Graz',
        'ShipRegion': null,
        'ShipCountry': 'Austria'
    },
    {
        'OrderID': 10259,
        'CustomerID': 'CENTC',
        'OrderDate': '1996-07-18T00:00:00.000Z',
        'ShippedDate': '1996-07-25T00:00:00.000Z',
        'Freight': 3.25,
        'ShipName': 'Centro comercial Moctezuma',
        'ShipAddress': 'Sierras de Granada 9993',
        'ShipCity': 'México D.F.',
        'ShipRegion': null,
        'ShipCountry': 'Mexico'
    },
    {
        'OrderID': 10260,
        'CustomerID': 'OTTIK',
        'OrderDate': '1996-07-19T00:00:00.000Z',
        'ShippedDate': '1996-07-29T00:00:00.000Z',
        'Freight': 55.09,
        'ShipName': 'Ottilies Käseladen',
        'ShipAddress': 'Mehrheimerstr. 369',
        'ShipCity': 'Köln',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10261,
        'CustomerID': 'QUEDE',
        'OrderDate': '1996-07-19T00:00:00.000Z',
        'ShippedDate': '1996-07-30T00:00:00.000Z',
        'Freight': 3.05,
        'ShipName': 'Que Delícia',
        'ShipAddress': 'Rua da Panificadora, 12',
        'ShipCity': 'Rio de Janeiro',
        'ShipRegion': 'RJ',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10262,
        'CustomerID': 'RATTC',
        'OrderDate': '1996-07-22T00:00:00.000Z',
        'ShippedDate': '1996-07-25T00:00:00.000Z',
        'Freight': 48.29,
        'ShipName': 'Rattlesnake Canyon Grocery',
        'ShipAddress': '2817 Milton Dr.',
        'ShipCity': 'Albuquerque',
        'ShipRegion': 'NM',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10263,
        'CustomerID': 'ERNSH',
        'OrderDate': '1996-07-23T00:00:00.000Z',
        'ShippedDate': '1996-07-31T00:00:00.000Z',

```



```

    'Freight': 146.06,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10264,
    'CustomerID': 'FOLKO',
    'OrderDate': '1996-07-24T00:00:00.000Z',
    'ShippedDate': '1996-08-23T00:00:00.000Z',
    'Freight': 3.67,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10265,
    'CustomerID': 'BLONP',
    'OrderDate': '1996-07-25T00:00:00.000Z',
    'ShippedDate': '1996-08-12T00:00:00.000Z',
    'Freight': 55.28,
    'ShipName': 'Blondel père et fils',
    'ShipAddress': '24, place Kléber',
    'ShipCity': 'Strasbourg',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10266,
    'CustomerID': 'WARTH',
    'OrderDate': '1996-07-26T00:00:00.000Z',
    'ShippedDate': '1996-07-31T00:00:00.000Z',
    'Freight': 25.73,
    'ShipName': 'Wartian Herkku',
    'ShipAddress': 'Torikatu 38',
    'ShipCity': 'Oulu',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10267,
    'CustomerID': 'FRANK',
    'OrderDate': '1996-07-29T00:00:00.000Z',
    'ShippedDate': '1996-08-06T00:00:00.000Z',
    'Freight': 208.58,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10268,

```

```

    'CustomerID': 'GROSR',
    'OrderDate': '1996-07-30T00:00:00.000Z',
    'ShippedDate': '1996-08-02T00:00:00.000Z',
    'Freight': 66.29,
    'ShipName': 'GROSELLA-Restaurante',
    'ShipAddress': '5a Ave. Los Palos Grandes',
    'ShipCity': 'Caracas',
    'ShipRegion': 'DF',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10269,
    'CustomerID': 'WHITC',
    'OrderDate': '1996-07-31T00:00:00.000Z',
    'ShippedDate': '1996-08-09T00:00:00.000Z',
    'Freight': 4.56,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10270,
    'CustomerID': 'WARTH',
    'OrderDate': '1996-08-01T00:00:00.000Z',
    'ShippedDate': '1996-08-02T00:00:00.000Z',
    'Freight': 136.54,
    'ShipName': 'Wartian Herkku',
    'ShipAddress': 'Torikatu 38',
    'ShipCity': 'Oulu',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10271,
    'CustomerID': 'SPLIR',
    'OrderDate': '1996-08-01T00:00:00.000Z',
    'ShippedDate': '1996-08-30T00:00:00.000Z',
    'Freight': 4.54,
    'ShipName': 'Split Rail Beer & Ale',
    'ShipAddress': 'P.O. Box 555',
    'ShipCity': 'Lander',
    'ShipRegion': 'WY',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10272,
    'CustomerID': 'RATTC',
    'OrderDate': '1996-08-02T00:00:00.000Z',
    'ShippedDate': '1996-08-06T00:00:00.000Z',
    'Freight': 98.03,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  }

```

```

    },
    {
        'OrderID': 10273,
        'CustomerID': 'QUICK',
        'OrderDate': '1996-08-05T00:00:00.000Z',
        'ShippedDate': '1996-08-12T00:00:00.000Z',
        'Freight': 76.07,
        'ShipName': 'QUICK-Stop',
        'ShipAddress': 'Taucherstraße 10',
        'ShipCity': 'Cunewalde',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10274,
        'CustomerID': 'VINET',
        'OrderDate': '1996-08-06T00:00:00.000Z',
        'ShippedDate': '1996-08-16T00:00:00.000Z',
        'Freight': 6.01,
        'ShipName': 'Vins et alcools Chevalier',
        'ShipAddress': '59 rue de l\'Abbaye',
        'ShipCity': 'Reims',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10275,
        'CustomerID': 'MAGAA',
        'OrderDate': '1996-08-07T00:00:00.000Z',
        'ShippedDate': '1996-08-09T00:00:00.000Z',
        'Freight': 26.93,
        'ShipName': 'Magazzini Alimentari Riuniti',
        'ShipAddress': 'Via Ludovico il Moro 22',
        'ShipCity': 'Bergamo',
        'ShipRegion': null,
        'ShipCountry': 'Italy'
    },
    {
        'OrderID': 10276,
        'CustomerID': 'TORTU',
        'OrderDate': '1996-08-08T00:00:00.000Z',
        'ShippedDate': '1996-08-14T00:00:00.000Z',
        'Freight': 13.84,
        'ShipName': 'Tortuga Restaurante',
        'ShipAddress': 'Avda. Azteca 123',
        'ShipCity': 'México D.F.',
        'ShipRegion': null,
        'ShipCountry': 'Mexico'
    },
    {
        'OrderID': 10277,
        'CustomerID': 'MORGK',
        'OrderDate': '1996-08-09T00:00:00.000Z',
        'ShippedDate': '1996-08-13T00:00:00.000Z',
        'Freight': 125.77,
        'ShipName': 'Morgenstern Gesundkost',
        'ShipAddress': 'Heerstr. 22',
    }

```

```

        'ShipCity': 'Leipzig',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10278,
        'CustomerID': 'BERGS',
        'OrderDate': '1996-08-12T00:00:00.000Z',
        'ShippedDate': '1996-08-16T00:00:00.000Z',
        'Freight': 92.69,
        'ShipName': 'Berglunds snabbköp',
        'ShipAddress': 'Berguvsvägen 8',
        'ShipCity': 'Luleå',
        'ShipRegion': null,
        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 10279,
        'CustomerID': 'LEHMS',
        'OrderDate': '1996-08-13T00:00:00.000Z',
        'ShippedDate': '1996-08-16T00:00:00.000Z',
        'Freight': 25.83,
        'ShipName': 'Lehmanns Marktstand',
        'ShipAddress': 'Magazinweg 7',
        'ShipCity': 'Frankfurt a.M.',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10280,
        'CustomerID': 'BERGS',
        'OrderDate': '1996-08-14T00:00:00.000Z',
        'ShippedDate': '1996-09-12T00:00:00.000Z',
        'Freight': 8.98,
        'ShipName': 'Berglunds snabbköp',
        'ShipAddress': 'Berguvsvägen 8',
        'ShipCity': 'Luleå',
        'ShipRegion': null,
        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 10281,
        'CustomerID': 'ROMEY',
        'OrderDate': '1996-08-14T00:00:00.000Z',
        'ShippedDate': '1996-08-21T00:00:00.000Z',
        'Freight': 2.94,
        'ShipName': 'Romero y tomillo',
        'ShipAddress': 'Gran Vía, 1',
        'ShipCity': 'Madrid',
        'ShipRegion': null,
        'ShipCountry': 'Spain'
    },
    {
        'OrderID': 10282,
        'CustomerID': 'ROMEY',
        'OrderDate': '1996-08-15T00:00:00.000Z',
        'ShippedDate': '1996-08-21T00:00:00.000Z',

```

```

    'Freight': 12.69,
    'ShipName': 'Romero y tomillo',
    'ShipAddress': 'Gran Vía, 1',
    'ShipCity': 'Madrid',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10283,
    'CustomerID': 'LILAS',
    'OrderDate': '1996-08-16T00:00:00.000Z',
    'ShippedDate': '1996-08-23T00:00:00.000Z',
    'Freight': 84.81,
    'ShipName': 'LILA-Supermercado',
    'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10284,
    'CustomerID': 'LEHMS',
    'OrderDate': '1996-08-19T00:00:00.000Z',
    'ShippedDate': '1996-08-27T00:00:00.000Z',
    'Freight': 76.56,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10285,
    'CustomerID': 'QUICK',
    'OrderDate': '1996-08-20T00:00:00.000Z',
    'ShippedDate': '1996-08-26T00:00:00.000Z',
    'Freight': 76.83,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10286,
    'CustomerID': 'QUICK',
    'OrderDate': '1996-08-21T00:00:00.000Z',
    'ShippedDate': '1996-08-30T00:00:00.000Z',
    'Freight': 229.24,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10287,

```

```

    'CustomerID': 'RICAR',
    'OrderDate': '1996-08-22T00:00:00.000Z',
    'ShippedDate': '1996-08-28T00:00:00.000Z',
    'Freight': 12.76,
    'ShipName': 'Ricardo Adocicados',
    'ShipAddress': 'Av. Copacabana, 267',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10288,
    'CustomerID': 'REGGC',
    'OrderDate': '1996-08-23T00:00:00.000Z',
    'ShippedDate': '1996-09-03T00:00:00.000Z',
    'Freight': 7.45,
    'ShipName': 'Reggiani Caseifici',
    'ShipAddress': 'Strada Provinciale 124',
    'ShipCity': 'Reggio Emilia',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10289,
    'CustomerID': 'BSBEV',
    'OrderDate': '1996-08-26T00:00:00.000Z',
    'ShippedDate': '1996-08-28T00:00:00.000Z',
    'Freight': 22.77,
    'ShipName': 'B\ Beverages',
    'ShipAddress': 'Fauntleroy Circus',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10290,
    'CustomerID': 'COMMI',
    'OrderDate': '1996-08-27T00:00:00.000Z',
    'ShippedDate': '1996-09-03T00:00:00.000Z',
    'Freight': 79.7,
    'ShipName': 'Comércio Mineiro',
    'ShipAddress': 'Av. dos Lusíadas, 23',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10291,
    'CustomerID': 'QUEDE',
    'OrderDate': '1996-08-27T00:00:00.000Z',
    'ShippedDate': '1996-09-04T00:00:00.000Z',
    'Freight': 6.4,
    'ShipName': 'Que Delícia',
    'ShipAddress': 'Rua da Panificadora, 12',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  }

```

```

    },
    {
        'OrderID': 10292,
        'CustomerID': 'TRADH',
        'OrderDate': '1996-08-28T00:00:00.000Z',
        'ShippedDate': '1996-09-02T00:00:00.000Z',
        'Freight': 1.35,
        'ShipName': 'Tradiçao Hipermercados',
        'ShipAddress': 'Av. Inês de Castro, 414',
        'ShipCity': 'Sao Paulo',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10293,
        'CustomerID': 'TORTU',
        'OrderDate': '1996-08-29T00:00:00.000Z',
        'ShippedDate': '1996-09-11T00:00:00.000Z',
        'Freight': 21.18,
        'ShipName': 'Tortuga Restaurante',
        'ShipAddress': 'Avda. Azteca 123',
        'ShipCity': 'México D.F.',
        'ShipRegion': null,
        'ShipCountry': 'Mexico'
    },
    {
        'OrderID': 10294,
        'CustomerID': 'RATTC',
        'OrderDate': '1996-08-30T00:00:00.000Z',
        'ShippedDate': '1996-09-05T00:00:00.000Z',
        'Freight': 147.26,
        'ShipName': 'Rattlesnake Canyon Grocery',
        'ShipAddress': '2817 Milton Dr.',
        'ShipCity': 'Albuquerque',
        'ShipRegion': 'NM',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10295,
        'CustomerID': 'VINET',
        'OrderDate': '1996-09-02T00:00:00.000Z',
        'ShippedDate': '1996-09-10T00:00:00.000Z',
        'Freight': 1.15,
        'ShipName': 'Vins et alcools Chevalier',
        'ShipAddress': '59 rue de l\'Abbaye',
        'ShipCity': 'Reims',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10296,
        'CustomerID': 'LILAS',
        'OrderDate': '1996-09-03T00:00:00.000Z',
        'ShippedDate': '1996-09-11T00:00:00.000Z',
        'Freight': 0.12,
        'ShipName': 'LILA-Supermercado',
        'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',

```

```

    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10297,
    'CustomerID': 'BLONP',
    'OrderDate': '1996-09-04T00:00:00.000Z',
    'ShippedDate': '1996-09-10T00:00:00.000Z',
    'Freight': 5.74,
    'ShipName': 'Blondel père et fils',
    'ShipAddress': '24, place Kléber',
    'ShipCity': 'Strasbourg',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10298,
    'CustomerID': 'HUNGO',
    'OrderDate': '1996-09-05T00:00:00.000Z',
    'ShippedDate': '1996-09-11T00:00:00.000Z',
    'Freight': 168.22,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10299,
    'CustomerID': 'RICAR',
    'OrderDate': '1996-09-06T00:00:00.000Z',
    'ShippedDate': '1996-09-13T00:00:00.000Z',
    'Freight': 29.76,
    'ShipName': 'Ricardo Adocicados',
    'ShipAddress': 'Av. Copacabana, 267',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10300,
    'CustomerID': 'MAGAA',
    'OrderDate': '1996-09-09T00:00:00.000Z',
    'ShippedDate': '1996-09-18T00:00:00.000Z',
    'Freight': 17.68,
    'ShipName': 'Magazzini Alimentari Riuniti',
    'ShipAddress': 'Via Ludovico il Moro 22',
    'ShipCity': 'Bergamo',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10301,
    'CustomerID': 'WANDK',
    'OrderDate': '1996-09-09T00:00:00.000Z',
    'ShippedDate': '1996-09-17T00:00:00.000Z',

```



```

    'Freight': 45.08,
    'ShipName': 'Die Wandernde Kuh',
    'ShipAddress': 'Adenauerallee 900',
    'ShipCity': 'Stuttgart',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10302,
    'CustomerID': 'SUPRD',
    'OrderDate': '1996-09-10T00:00:00.000Z',
    'ShippedDate': '1996-10-09T00:00:00.000Z',
    'Freight': 6.27,
    'ShipName': 'Suprêmes délices',
    'ShipAddress': 'Boulevard Tirou, 255',
    'ShipCity': 'Charleroi',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10303,
    'CustomerID': 'GODOS',
    'OrderDate': '1996-09-11T00:00:00.000Z',
    'ShippedDate': '1996-09-18T00:00:00.000Z',
    'Freight': 107.83,
    'ShipName': 'Godos Cocina Típica',
    'ShipAddress': 'C/ Romero, 33',
    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10304,
    'CustomerID': 'TORTU',
    'OrderDate': '1996-09-12T00:00:00.000Z',
    'ShippedDate': '1996-09-17T00:00:00.000Z',
    'Freight': 63.79,
    'ShipName': 'Tortuga Restaurante',
    'ShipAddress': 'Avda. Azteca 123',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10305,
    'CustomerID': 'OLDWO',
    'OrderDate': '1996-09-13T00:00:00.000Z',
    'ShippedDate': '1996-10-09T00:00:00.000Z',
    'Freight': 257.62,
    'ShipName': 'Old World Delicatessen',
    'ShipAddress': '2743 Bering St.',
    'ShipCity': 'Anchorage',
    'ShipRegion': 'AK',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10306,

```

```

      'CustomerID': 'ROMEY',
      'OrderDate': '1996-09-16T00:00:00.000Z',
      'ShippedDate': '1996-09-23T00:00:00.000Z',
      'Freight': 7.56,
      'ShipName': 'Romero y tomillo',
      'ShipAddress': 'Gran Vía, 1',
      'ShipCity': 'Madrid',
      'ShipRegion': null,
      'ShipCountry': 'Spain'
    },
    {
      'OrderID': 10307,
      'CustomerID': 'LONEP',
      'OrderDate': '1996-09-17T00:00:00.000Z',
      'ShippedDate': '1996-09-25T00:00:00.000Z',
      'Freight': 0.56,
      'ShipName': 'Lonesome Pine Restaurant',
      'ShipAddress': '89 Chiaroscuro Rd.',
      'ShipCity': 'Portland',
      'ShipRegion': 'OR',
      'ShipCountry': 'USA'
    },
    {
      'OrderID': 10308,
      'CustomerID': 'ANATR',
      'OrderDate': '1996-09-18T00:00:00.000Z',
      'ShippedDate': '1996-09-24T00:00:00.000Z',
      'Freight': 1.61,
      'ShipName': 'Ana Trujillo Emparedados y helados',
      'ShipAddress': 'Avda. de la Constitución 2222',
      'ShipCity': 'México D.F.',
      'ShipRegion': null,
      'ShipCountry': 'Mexico'
    },
    {
      'OrderID': 10309,
      'CustomerID': 'HUNGO',
      'OrderDate': '1996-09-19T00:00:00.000Z',
      'ShippedDate': '1996-10-23T00:00:00.000Z',
      'Freight': 47.3,
      'ShipName': 'Hungry Owl All-Night Grocers',
      'ShipAddress': '8 Johnstown Road',
      'ShipCity': 'Cork',
      'ShipRegion': 'Co. Cork',
      'ShipCountry': 'Ireland'
    },
    {
      'OrderID': 10310,
      'CustomerID': 'THEBI',
      'OrderDate': '1996-09-20T00:00:00.000Z',
      'ShippedDate': '1996-09-27T00:00:00.000Z',
      'Freight': 17.52,
      'ShipName': 'The Big Cheese',
      'ShipAddress': '89 Jefferson Way Suite 2',
      'ShipCity': 'Portland',
      'ShipRegion': 'OR',
      'ShipCountry': 'USA'
    }
  ]

```

```

},
{
  'OrderID': 10311,
  'CustomerID': 'DUMON',
  'OrderDate': '1996-09-20T00:00:00.000Z',
  'ShippedDate': '1996-09-26T00:00:00.000Z',
  'Freight': 24.69,
  'ShipName': 'Du monde entier',
  'ShipAddress': '67, rue des Cinquante Otages',
  'ShipCity': 'Nantes',
  'ShipRegion': null,
  'ShipCountry': 'France'
},
{
  'OrderID': 10312,
  'CustomerID': 'WANDK',
  'OrderDate': '1996-09-23T00:00:00.000Z',
  'ShippedDate': '1996-10-03T00:00:00.000Z',
  'Freight': 40.26,
  'ShipName': 'Die Wandernde Kuh',
  'ShipAddress': 'Adenauerallee 900',
  'ShipCity': 'Stuttgart',
  'ShipRegion': null,
  'ShipCountry': 'Germany'
},
{
  'OrderID': 10313,
  'CustomerID': 'QUICK',
  'OrderDate': '1996-09-24T00:00:00.000Z',
  'ShippedDate': '1996-10-04T00:00:00.000Z',
  'Freight': 1.96,
  'ShipName': 'QUICK-Stop',
  'ShipAddress': 'Taucherstraße 10',
  'ShipCity': 'Cunewalde',
  'ShipRegion': null,
  'ShipCountry': 'Germany'
},
{
  'OrderID': 10314,
  'CustomerID': 'RATTC',
  'OrderDate': '1996-09-25T00:00:00.000Z',
  'ShippedDate': '1996-10-04T00:00:00.000Z',
  'Freight': 74.16,
  'ShipName': 'Rattlesnake Canyon Grocery',
  'ShipAddress': '2817 Milton Dr.',
  'ShipCity': 'Albuquerque',
  'ShipRegion': 'NM',
  'ShipCountry': 'USA'
},
{
  'OrderID': 10315,
  'CustomerID': 'ISLAT',
  'OrderDate': '1996-09-26T00:00:00.000Z',
  'ShippedDate': '1996-10-03T00:00:00.000Z',
  'Freight': 41.76,
  'ShipName': 'Island Trading',
  'ShipAddress': 'Garden House Crowther Way',

```

```

        'ShipCity': 'Cowes',
        'ShipRegion': 'Isle of Wight',
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10316,
        'CustomerID': 'RATTC',
        'OrderDate': '1996-09-27T00:00:00.000Z',
        'ShippedDate': '1996-10-08T00:00:00.000Z',
        'Freight': 150.15,
        'ShipName': 'Rattlesnake Canyon Grocery',
        'ShipAddress': '2817 Milton Dr.',
        'ShipCity': 'Albuquerque',
        'ShipRegion': 'NM',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10317,
        'CustomerID': 'LONEP',
        'OrderDate': '1996-09-30T00:00:00.000Z',
        'ShippedDate': '1996-10-10T00:00:00.000Z',
        'Freight': 12.69,
        'ShipName': 'Lonesome Pine Restaurant',
        'ShipAddress': '89 Chiaroscuro Rd.',
        'ShipCity': 'Portland',
        'ShipRegion': 'OR',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10318,
        'CustomerID': 'ISLAT',
        'OrderDate': '1996-10-01T00:00:00.000Z',
        'ShippedDate': '1996-10-04T00:00:00.000Z',
        'Freight': 4.73,
        'ShipName': 'Island Trading',
        'ShipAddress': 'Garden House Crowther Way',
        'ShipCity': 'Cowes',
        'ShipRegion': 'Isle of Wight',
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10319,
        'CustomerID': 'TORTU',
        'OrderDate': '1996-10-02T00:00:00.000Z',
        'ShippedDate': '1996-10-11T00:00:00.000Z',
        'Freight': 64.5,
        'ShipName': 'Tortuga Restaurante',
        'ShipAddress': 'Avda. Azteca 123',
        'ShipCity': 'México D.F.',
        'ShipRegion': null,
        'ShipCountry': 'Mexico'
    },
    {
        'OrderID': 10320,
        'CustomerID': 'WARTH',
        'OrderDate': '1996-10-03T00:00:00.000Z',
        'ShippedDate': '1996-10-18T00:00:00.000Z',

```

```

    'Freight': 34.57,
    'ShipName': 'Wartian Herkku',
    'ShipAddress': 'Torikatu 38',
    'ShipCity': 'Oulu',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10321,
    'CustomerID': 'ISLAT',
    'OrderDate': '1996-10-03T00:00:00.000Z',
    'ShippedDate': '1996-10-11T00:00:00.000Z',
    'Freight': 3.43,
    'ShipName': 'Island Trading',
    'ShipAddress': 'Garden House Crowther Way',
    'ShipCity': 'Cowes',
    'ShipRegion': 'Isle of Wight',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10322,
    'CustomerID': 'PERIC',
    'OrderDate': '1996-10-04T00:00:00.000Z',
    'ShippedDate': '1996-10-23T00:00:00.000Z',
    'Freight': 0.4,
    'ShipName': 'Pericles Comidas clásicas',
    'ShipAddress': 'Calle Dr. Jorge Cash 321',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10323,
    'CustomerID': 'KOENE',
    'OrderDate': '1996-10-07T00:00:00.000Z',
    'ShippedDate': '1996-10-14T00:00:00.000Z',
    'Freight': 4.88,
    'ShipName': 'Königlich Essen',
    'ShipAddress': 'Maubelstr. 90',
    'ShipCity': 'Brandenburg',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10324,
    'CustomerID': 'SAVEA',
    'OrderDate': '1996-10-08T00:00:00.000Z',
    'ShippedDate': '1996-10-10T00:00:00.000Z',
    'Freight': 214.27,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10325,

```

```

      'CustomerID': 'KOENE',
      'OrderDate': '1996-10-09T00:00:00.000Z',
      'ShippedDate': '1996-10-14T00:00:00.000Z',
      'Freight': 64.86,
      'ShipName': 'Königlich Essen',
      'ShipAddress': 'Maubelstr. 90',
      'ShipCity': 'Brandenburg',
      'ShipRegion': null,
      'ShipCountry': 'Germany'
    },
    {
      'OrderID': 10326,
      'CustomerID': 'BOLID',
      'OrderDate': '1996-10-10T00:00:00.000Z',
      'ShippedDate': '1996-10-14T00:00:00.000Z',
      'Freight': 77.92,
      'ShipName': 'Bólido Comidas preparadas',
      'ShipAddress': 'C/ Araquil, 67',
      'ShipCity': 'Madrid',
      'ShipRegion': null,
      'ShipCountry': 'Spain'
    },
    {
      'OrderID': 10327,
      'CustomerID': 'FOLKO',
      'OrderDate': '1996-10-11T00:00:00.000Z',
      'ShippedDate': '1996-10-14T00:00:00.000Z',
      'Freight': 63.36,
      'ShipName': 'Folk och fä HB',
      'ShipAddress': 'Åkergatan 24',
      'ShipCity': 'Bräcke',
      'ShipRegion': null,
      'ShipCountry': 'Sweden'
    },
    {
      'OrderID': 10328,
      'CustomerID': 'FURIB',
      'OrderDate': '1996-10-14T00:00:00.000Z',
      'ShippedDate': '1996-10-17T00:00:00.000Z',
      'Freight': 87.03,
      'ShipName': 'Furia Bacalhau e Frutos do Mar',
      'ShipAddress': 'Jardim das rosas n. 32',
      'ShipCity': 'Lisboa',
      'ShipRegion': null,
      'ShipCountry': 'Portugal'
    },
    {
      'OrderID': 10329,
      'CustomerID': 'SPLIR',
      'OrderDate': '1996-10-15T00:00:00.000Z',
      'ShippedDate': '1996-10-23T00:00:00.000Z',
      'Freight': 191.67,
      'ShipName': 'Split Rail Beer & Ale',
      'ShipAddress': 'P.O. Box 555',
      'ShipCity': 'Lander',
      'ShipRegion': 'WY',
      'ShipCountry': 'USA'
    }
  ]

```

```

},
{
  'OrderID': 10330,
  'CustomerID': 'LILAS',
  'OrderDate': '1996-10-16T00:00:00.000Z',
  'ShippedDate': '1996-10-28T00:00:00.000Z',
  'Freight': 12.75,
  'ShipName': 'LILA-Supermercado',
  'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
  'ShipCity': 'Barquisimeto',
  'ShipRegion': 'Lara',
  'ShipCountry': 'Venezuela'
},
{
  'OrderID': 10331,
  'CustomerID': 'BONAP',
  'OrderDate': '1996-10-16T00:00:00.000Z',
  'ShippedDate': '1996-10-21T00:00:00.000Z',
  'Freight': 10.19,
  'ShipName': 'Bon app',
  'ShipAddress': '12, rue des Bouchers',
  'ShipCity': 'Marseille',
  'ShipRegion': null,
  'ShipCountry': 'France'
},
{
  'OrderID': 10332,
  'CustomerID': 'MEREP',
  'OrderDate': '1996-10-17T00:00:00.000Z',
  'ShippedDate': '1996-10-21T00:00:00.000Z',
  'Freight': 52.84,
  'ShipName': 'Mère Paillarde',
  'ShipAddress': '43 rue St. Laurent',
  'ShipCity': 'Montréal',
  'ShipRegion': 'Québec',
  'ShipCountry': 'Canada'
},
{
  'OrderID': 10333,
  'CustomerID': 'WARTH',
  'OrderDate': '1996-10-18T00:00:00.000Z',
  'ShippedDate': '1996-10-25T00:00:00.000Z',
  'Freight': 0.59,
  'ShipName': 'Wartian Herkku',
  'ShipAddress': 'Torikatu 38',
  'ShipCity': 'Oulu',
  'ShipRegion': null,
  'ShipCountry': 'Finland'
},
{
  'OrderID': 10334,
  'CustomerID': 'VICTE',
  'OrderDate': '1996-10-21T00:00:00.000Z',
  'ShippedDate': '1996-10-28T00:00:00.000Z',
  'Freight': 8.56,
  'ShipName': 'Victuailles en stock',
  'ShipAddress': '2, rue du Commerce',

```

```

        'ShipCity': 'Lyon',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10335,
        'CustomerID': 'HUNGO',
        'OrderDate': '1996-10-22T00:00:00.000Z',
        'ShippedDate': '1996-10-24T00:00:00.000Z',
        'Freight': 42.11,
        'ShipName': 'Hungry Owl All-Night Grocers',
        'ShipAddress': '8 Johnstown Road',
        'ShipCity': 'Cork',
        'ShipRegion': 'Co. Cork',
        'ShipCountry': 'Ireland'
    },
    {
        'OrderID': 10336,
        'CustomerID': 'PRINI',
        'OrderDate': '1996-10-23T00:00:00.000Z',
        'ShippedDate': '1996-10-25T00:00:00.000Z',
        'Freight': 15.51,
        'ShipName': 'Princesa Isabel Vinhos',
        'ShipAddress': 'Estrada da saúde n. 58',
        'ShipCity': 'Lisboa',
        'ShipRegion': null,
        'ShipCountry': 'Portugal'
    },
    {
        'OrderID': 10337,
        'CustomerID': 'FRANK',
        'OrderDate': '1996-10-24T00:00:00.000Z',
        'ShippedDate': '1996-10-29T00:00:00.000Z',
        'Freight': 108.26,
        'ShipName': 'Frankenversand',
        'ShipAddress': 'Berliner Platz 43',
        'ShipCity': 'München',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10338,
        'CustomerID': 'OLDWO',
        'OrderDate': '1996-10-25T00:00:00.000Z',
        'ShippedDate': '1996-10-29T00:00:00.000Z',
        'Freight': 84.21,
        'ShipName': 'Old World Delicatessen',
        'ShipAddress': '2743 Bering St.',
        'ShipCity': 'Anchorage',
        'ShipRegion': 'AK',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10339,
        'CustomerID': 'MEREP',
        'OrderDate': '1996-10-28T00:00:00.000Z',
        'ShippedDate': '1996-11-04T00:00:00.000Z',

```



```

    'Freight': 15.66,
    'ShipName': 'Mère Paillarde',
    'ShipAddress': '43 rue St. Laurent',
    'ShipCity': 'Montréal',
    'ShipRegion': 'Québec',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10340,
    'CustomerID': 'BONAP',
    'OrderDate': '1996-10-29T00:00:00.000Z',
    'ShippedDate': '1996-11-08T00:00:00.000Z',
    'Freight': 166.31,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10341,
    'CustomerID': 'SIMOB',
    'OrderDate': '1996-10-29T00:00:00.000Z',
    'ShippedDate': '1996-11-05T00:00:00.000Z',
    'Freight': 26.78,
    'ShipName': 'Simons bistro',
    'ShipAddress': 'Vinbæltet 34',
    'ShipCity': 'Kobenhavn',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10342,
    'CustomerID': 'FRANK',
    'OrderDate': '1996-10-30T00:00:00.000Z',
    'ShippedDate': '1996-11-04T00:00:00.000Z',
    'Freight': 54.83,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10343,
    'CustomerID': 'LEHMS',
    'OrderDate': '1996-10-31T00:00:00.000Z',
    'ShippedDate': '1996-11-06T00:00:00.000Z',
    'Freight': 110.37,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10344,

```

```

    'CustomerID': 'WHITC',
    'OrderDate': '1996-11-01T00:00:00.000Z',
    'ShippedDate': '1996-11-05T00:00:00.000Z',
    'Freight': 23.29,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10345,
    'CustomerID': 'QUICK',
    'OrderDate': '1996-11-04T00:00:00.000Z',
    'ShippedDate': '1996-11-11T00:00:00.000Z',
    'Freight': 249.06,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10346,
    'CustomerID': 'RATTC',
    'OrderDate': '1996-11-05T00:00:00.000Z',
    'ShippedDate': '1996-11-08T00:00:00.000Z',
    'Freight': 142.08,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10347,
    'CustomerID': 'FAMIA',
    'OrderDate': '1996-11-06T00:00:00.000Z',
    'ShippedDate': '1996-11-08T00:00:00.000Z',
    'Freight': 3.1,
    'ShipName': 'Familia Arquibaldo',
    'ShipAddress': 'Rua Orós, 92',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10348,
    'CustomerID': 'WANDK',
    'OrderDate': '1996-11-07T00:00:00.000Z',
    'ShippedDate': '1996-11-15T00:00:00.000Z',
    'Freight': 0.78,
    'ShipName': 'Die Wandernde Kuh',
    'ShipAddress': 'Adenauerallee 900',
    'ShipCity': 'Stuttgart',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  }

```

```

},
{
  'OrderID': 10349,
  'CustomerID': 'SPLIR',
  'OrderDate': '1996-11-08T00:00:00.000Z',
  'ShippedDate': '1996-11-15T00:00:00.000Z',
  'Freight': 8.63,
  'ShipName': 'Split Rail Beer & Ale',
  'ShipAddress': 'P.O. Box 555',
  'ShipCity': 'Lander',
  'ShipRegion': 'WY',
  'ShipCountry': 'USA'
},
{
  'OrderID': 10350,
  'CustomerID': 'LAMAI',
  'OrderDate': '1996-11-11T00:00:00.000Z',
  'ShippedDate': '1996-12-03T00:00:00.000Z',
  'Freight': 64.19,
  'ShipName': 'La maison d\'Asie',
  'ShipAddress': '1 rue Alsace-Lorraine',
  'ShipCity': 'Toulouse',
  'ShipRegion': null,
  'ShipCountry': 'France'
},
{
  'OrderID': 10351,
  'CustomerID': 'ERNSH',
  'OrderDate': '1996-11-11T00:00:00.000Z',
  'ShippedDate': '1996-11-20T00:00:00.000Z',
  'Freight': 162.33,
  'ShipName': 'Ernst Handel',
  'ShipAddress': 'Kirchgasse 6',
  'ShipCity': 'Graz',
  'ShipRegion': null,
  'ShipCountry': 'Austria'
},
{
  'OrderID': 10352,
  'CustomerID': 'FURIB',
  'OrderDate': '1996-11-12T00:00:00.000Z',
  'ShippedDate': '1996-11-18T00:00:00.000Z',
  'Freight': 1.3,
  'ShipName': 'Furia Bacalhau e Frutos do Mar',
  'ShipAddress': 'Jardim das rosas n. 32',
  'ShipCity': 'Lisboa',
  'ShipRegion': null,
  'ShipCountry': 'Portugal'
},
{
  'OrderID': 10353,
  'CustomerID': 'PICCO',
  'OrderDate': '1996-11-13T00:00:00.000Z',
  'ShippedDate': '1996-11-25T00:00:00.000Z',
  'Freight': 360.63,
  'ShipName': 'Piccolo und mehr',
  'ShipAddress': 'Geislweg 14',

```

```

        'ShipCity': 'Salzburg',
        'ShipRegion': null,
        'ShipCountry': 'Austria'
    },
    {
        'OrderID': 10354,
        'CustomerID': 'PERIC',
        'OrderDate': '1996-11-14T00:00:00.000Z',
        'ShippedDate': '1996-11-20T00:00:00.000Z',
        'Freight': 53.8,
        'ShipName': 'Pericles Comidas clásicas',
        'ShipAddress': 'Calle Dr. Jorge Cash 321',
        'ShipCity': 'México D.F.',
        'ShipRegion': null,
        'ShipCountry': 'Mexico'
    },
    {
        'OrderID': 10355,
        'CustomerID': 'AROUT',
        'OrderDate': '1996-11-15T00:00:00.000Z',
        'ShippedDate': '1996-11-20T00:00:00.000Z',
        'Freight': 41.95,
        'ShipName': 'Around the Horn',
        'ShipAddress': 'Brook Farm Stratford St. Mary',
        'ShipCity': 'Colchester',
        'ShipRegion': 'Essex',
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10356,
        'CustomerID': 'WANDK',
        'OrderDate': '1996-11-18T00:00:00.000Z',
        'ShippedDate': '1996-11-27T00:00:00.000Z',
        'Freight': 36.71,
        'ShipName': 'Die Wandernde Kuh',
        'ShipAddress': 'Adenauerallee 900',
        'ShipCity': 'Stuttgart',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10357,
        'CustomerID': 'LILAS',
        'OrderDate': '1996-11-19T00:00:00.000Z',
        'ShippedDate': '1996-12-02T00:00:00.000Z',
        'Freight': 34.88,
        'ShipName': 'LILA-Supermercado',
        'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
        'ShipCity': 'Barquisimeto',
        'ShipRegion': 'Lara',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10358,
        'CustomerID': 'LAMAI',
        'OrderDate': '1996-11-20T00:00:00.000Z',
        'ShippedDate': '1996-11-27T00:00:00.000Z',

```

```

    'Freight': 19.64,
    'ShipName': 'La maison d\'Asie',
    'ShipAddress': '1 rue Alsace-Lorraine',
    'ShipCity': 'Toulouse',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10359,
    'CustomerID': 'SEVES',
    'OrderDate': '1996-11-21T00:00:00.000Z',
    'ShippedDate': '1996-11-26T00:00:00.000Z',
    'Freight': 288.43,
    'ShipName': 'Seven Seas Imports',
    'ShipAddress': '90 Wadhurst Rd.',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10360,
    'CustomerID': 'BLONP',
    'OrderDate': '1996-11-22T00:00:00.000Z',
    'ShippedDate': '1996-12-02T00:00:00.000Z',
    'Freight': 131.7,
    'ShipName': 'Blondel père et fils',
    'ShipAddress': '24, place Kléber',
    'ShipCity': 'Strasbourg',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10361,
    'CustomerID': 'QUICK',
    'OrderDate': '1996-11-22T00:00:00.000Z',
    'ShippedDate': '1996-12-03T00:00:00.000Z',
    'Freight': 183.17,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10362,
    'CustomerID': 'BONAP',
    'OrderDate': '1996-11-25T00:00:00.000Z',
    'ShippedDate': '1996-11-28T00:00:00.000Z',
    'Freight': 96.04,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10363,

```

```

        'CustomerID': 'DRACD',
        'OrderDate': '1996-11-26T00:00:00.000Z',
        'ShippedDate': '1996-12-04T00:00:00.000Z',
        'Freight': 30.54,
        'ShipName': 'Drachenblut Delikatessen',
        'ShipAddress': 'Walserweg 21',
        'ShipCity': 'Aachen',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10364,
        'CustomerID': 'EASTC',
        'OrderDate': '1996-11-26T00:00:00.000Z',
        'ShippedDate': '1996-12-04T00:00:00.000Z',
        'Freight': 71.97,
        'ShipName': 'Eastern Connection',
        'ShipAddress': '35 King George',
        'ShipCity': 'London',
        'ShipRegion': null,
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10365,
        'CustomerID': 'ANTON',
        'OrderDate': '1996-11-27T00:00:00.000Z',
        'ShippedDate': '1996-12-02T00:00:00.000Z',
        'Freight': 22,
        'ShipName': 'Antonio Moreno Taquería',
        'ShipAddress': 'Mataderos 2312',
        'ShipCity': 'México D.F.',
        'ShipRegion': null,
        'ShipCountry': 'Mexico'
    },
    {
        'OrderID': 10366,
        'CustomerID': 'GALED',
        'OrderDate': '1996-11-28T00:00:00.000Z',
        'ShippedDate': '1996-12-30T00:00:00.000Z',
        'Freight': 10.14,
        'ShipName': 'Galería del gastronómo',
        'ShipAddress': 'Rambla de Cataluña, 23',
        'ShipCity': 'Barcelona',
        'ShipRegion': null,
        'ShipCountry': 'Spain'
    },
    {
        'OrderID': 10367,
        'CustomerID': 'VAFFE',
        'OrderDate': '1996-11-28T00:00:00.000Z',
        'ShippedDate': '1996-12-02T00:00:00.000Z',
        'Freight': 13.55,
        'ShipName': 'Vaffeljernet',
        'ShipAddress': 'Smagsloget 45',
        'ShipCity': 'Århus',
        'ShipRegion': null,
        'ShipCountry': 'Denmark'
    }

```

```

    },
    {
        'OrderID': 10368,
        'CustomerID': 'ERNSH',
        'OrderDate': '1996-11-29T00:00:00.000Z',
        'ShippedDate': '1996-12-02T00:00:00.000Z',
        'Freight': 101.95,
        'ShipName': 'Ernst Handel',
        'ShipAddress': 'Kirchgasse 6',
        'ShipCity': 'Graz',
        'ShipRegion': null,
        'ShipCountry': 'Austria'
    },
    {
        'OrderID': 10369,
        'CustomerID': 'SPLIR',
        'OrderDate': '1996-12-02T00:00:00.000Z',
        'ShippedDate': '1996-12-09T00:00:00.000Z',
        'Freight': 195.68,
        'ShipName': 'Split Rail Beer & Ale',
        'ShipAddress': 'P.O. Box 555',
        'ShipCity': 'Lander',
        'ShipRegion': 'WY',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10370,
        'CustomerID': 'CHOPS',
        'OrderDate': '1996-12-03T00:00:00.000Z',
        'ShippedDate': '1996-12-27T00:00:00.000Z',
        'Freight': 1.17,
        'ShipName': 'Chop-suey Chinese',
        'ShipAddress': 'Hauptstr. 31',
        'ShipCity': 'Bern',
        'ShipRegion': null,
        'ShipCountry': 'Switzerland'
    },
    {
        'OrderID': 10371,
        'CustomerID': 'LAMAI',
        'OrderDate': '1996-12-03T00:00:00.000Z',
        'ShippedDate': '1996-12-24T00:00:00.000Z',
        'Freight': 0.45,
        'ShipName': 'La maison d\'Asie',
        'ShipAddress': '1 rue Alsace-Lorraine',
        'ShipCity': 'Toulouse',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10372,
        'CustomerID': 'QUEEN',
        'OrderDate': '1996-12-04T00:00:00.000Z',
        'ShippedDate': '1996-12-09T00:00:00.000Z',
        'Freight': 890.78,
        'ShipName': 'Queen Cozinha',
        'ShipAddress': 'Alameda dos Canários, 891',
    }

```

```

        'ShipCity': 'Sao Paulo',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10373,
        'CustomerID': 'HUNGO',
        'OrderDate': '1996-12-05T00:00:00.000Z',
        'ShippedDate': '1996-12-11T00:00:00.000Z',
        'Freight': 124.12,
        'ShipName': 'Hungry Owl All-Night Grocers',
        'ShipAddress': '8 Johnstown Road',
        'ShipCity': 'Cork',
        'ShipRegion': 'Co. Cork',
        'ShipCountry': 'Ireland'
    },
    {
        'OrderID': 10374,
        'CustomerID': 'WOLZA',
        'OrderDate': '1996-12-05T00:00:00.000Z',
        'ShippedDate': '1996-12-09T00:00:00.000Z',
        'Freight': 3.94,
        'ShipName': 'Wolski Zajazd',
        'ShipAddress': 'ul. Filtrowa 68',
        'ShipCity': 'Warszawa',
        'ShipRegion': null,
        'ShipCountry': 'Poland'
    },
    {
        'OrderID': 10375,
        'CustomerID': 'HUNGC',
        'OrderDate': '1996-12-06T00:00:00.000Z',
        'ShippedDate': '1996-12-09T00:00:00.000Z',
        'Freight': 20.12,
        'ShipName': 'Hungry Coyote Import Store',
        'ShipAddress': 'City Center Plaza 516 Main St.',
        'ShipCity': 'Elgin',
        'ShipRegion': 'OR',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10376,
        'CustomerID': 'MEREP',
        'OrderDate': '1996-12-09T00:00:00.000Z',
        'ShippedDate': '1996-12-13T00:00:00.000Z',
        'Freight': 20.39,
        'ShipName': 'Mère Paillarde',
        'ShipAddress': '43 rue St. Laurent',
        'ShipCity': 'Montréal',
        'ShipRegion': 'Québec',
        'ShipCountry': 'Canada'
    },
    {
        'OrderID': 10377,
        'CustomerID': 'SEVES',
        'OrderDate': '1996-12-09T00:00:00.000Z',
        'ShippedDate': '1996-12-13T00:00:00.000Z',

```



```

    'Freight': 22.21,
    'ShipName': 'Seven Seas Imports',
    'ShipAddress': '90 Wadhurst Rd.',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10378,
    'CustomerID': 'FOLKO',
    'OrderDate': '1996-12-10T00:00:00.000Z',
    'ShippedDate': '1996-12-19T00:00:00.000Z',
    'Freight': 5.44,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10379,
    'CustomerID': 'QUEDE',
    'OrderDate': '1996-12-11T00:00:00.000Z',
    'ShippedDate': '1996-12-13T00:00:00.000Z',
    'Freight': 45.03,
    'ShipName': 'Que Delícia',
    'ShipAddress': 'Rua da Panificadora, 12',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10380,
    'CustomerID': 'HUNGO',
    'OrderDate': '1996-12-12T00:00:00.000Z',
    'ShippedDate': '1997-01-16T00:00:00.000Z',
    'Freight': 35.03,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10381,
    'CustomerID': 'LILAS',
    'OrderDate': '1996-12-12T00:00:00.000Z',
    'ShippedDate': '1996-12-13T00:00:00.000Z',
    'Freight': 7.99,
    'ShipName': 'LILA-Supermercado',
    'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10382,

```

```

    'CustomerID': 'ERNSH',
    'OrderDate': '1996-12-13T00:00:00.000Z',
    'ShippedDate': '1996-12-16T00:00:00.000Z',
    'Freight': 94.77,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10383,
    'CustomerID': 'AROUT',
    'OrderDate': '1996-12-16T00:00:00.000Z',
    'ShippedDate': '1996-12-18T00:00:00.000Z',
    'Freight': 34.24,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10384,
    'CustomerID': 'BERGS',
    'OrderDate': '1996-12-16T00:00:00.000Z',
    'ShippedDate': '1996-12-20T00:00:00.000Z',
    'Freight': 168.64,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10385,
    'CustomerID': 'SPLIR',
    'OrderDate': '1996-12-17T00:00:00.000Z',
    'ShippedDate': '1996-12-23T00:00:00.000Z',
    'Freight': 30.96,
    'ShipName': 'Split Rail Beer & Ale',
    'ShipAddress': 'P.O. Box 555',
    'ShipCity': 'Lander',
    'ShipRegion': 'WY',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10386,
    'CustomerID': 'FAMIA',
    'OrderDate': '1996-12-18T00:00:00.000Z',
    'ShippedDate': '1996-12-25T00:00:00.000Z',
    'Freight': 13.99,
    'ShipName': 'Familia Arquibaldo',
    'ShipAddress': 'Rua Orós, 92',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  }

```

```

    },
    {
        'OrderID': 10387,
        'CustomerID': 'SANTG',
        'OrderDate': '1996-12-18T00:00:00.000Z',
        'ShippedDate': '1996-12-20T00:00:00.000Z',
        'Freight': 93.63,
        'ShipName': 'Santé Gourmet',
        'ShipAddress': 'Erling Skakkes gate 78',
        'ShipCity': 'Stavern',
        'ShipRegion': null,
        'ShipCountry': 'Norway'
    },
    {
        'OrderID': 10388,
        'CustomerID': 'SEVES',
        'OrderDate': '1996-12-19T00:00:00.000Z',
        'ShippedDate': '1996-12-20T00:00:00.000Z',
        'Freight': 34.86,
        'ShipName': 'Seven Seas Imports',
        'ShipAddress': '90 Wadhurst Rd.',
        'ShipCity': 'London',
        'ShipRegion': null,
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10389,
        'CustomerID': 'BOTTM',
        'OrderDate': '1996-12-20T00:00:00.000Z',
        'ShippedDate': '1996-12-24T00:00:00.000Z',
        'Freight': 47.42,
        'ShipName': 'Bottom-Dollar Markets',
        'ShipAddress': '23 Tsawassen Blvd.',
        'ShipCity': 'Tsawassen',
        'ShipRegion': 'BC',
        'ShipCountry': 'Canada'
    },
    {
        'OrderID': 10390,
        'CustomerID': 'ERNSH',
        'OrderDate': '1996-12-23T00:00:00.000Z',
        'ShippedDate': '1996-12-26T00:00:00.000Z',
        'Freight': 126.38,
        'ShipName': 'Ernst Handel',
        'ShipAddress': 'Kirchgasse 6',
        'ShipCity': 'Graz',
        'ShipRegion': null,
        'ShipCountry': 'Austria'
    },
    {
        'OrderID': 10391,
        'CustomerID': 'DRACD',
        'OrderDate': '1996-12-23T00:00:00.000Z',
        'ShippedDate': '1996-12-31T00:00:00.000Z',
        'Freight': 5.45,
        'ShipName': 'Drachenblut Delikatessen',
        'ShipAddress': 'Walserweg 21',
    }

```

```

        'ShipCity': 'Aachen',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10392,
        'CustomerID': 'PICCO',
        'OrderDate': '1996-12-24T00:00:00.000Z',
        'ShippedDate': '1997-01-01T00:00:00.000Z',
        'Freight': 122.46,
        'ShipName': 'Piccolo und mehr',
        'ShipAddress': 'Geislweg 14',
        'ShipCity': 'Salzburg',
        'ShipRegion': null,
        'ShipCountry': 'Austria'
    },
    {
        'OrderID': 10393,
        'CustomerID': 'SAVEA',
        'OrderDate': '1996-12-25T00:00:00.000Z',
        'ShippedDate': '1997-01-03T00:00:00.000Z',
        'Freight': 126.56,
        'ShipName': 'Save-a-lot Markets',
        'ShipAddress': '187 Suffolk Ln.',
        'ShipCity': 'Boise',
        'ShipRegion': 'ID',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10394,
        'CustomerID': 'HUNGC',
        'OrderDate': '1996-12-25T00:00:00.000Z',
        'ShippedDate': '1997-01-03T00:00:00.000Z',
        'Freight': 30.34,
        'ShipName': 'Hungry Coyote Import Store',
        'ShipAddress': 'City Center Plaza 516 Main St.',
        'ShipCity': 'Elgin',
        'ShipRegion': 'OR',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10395,
        'CustomerID': 'HILAA',
        'OrderDate': '1996-12-26T00:00:00.000Z',
        'ShippedDate': '1997-01-03T00:00:00.000Z',
        'Freight': 184.41,
        'ShipName': 'HILARION-Abastos',
        'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
        'ShipCity': 'San Cristóbal',
        'ShipRegion': 'Táchira',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10396,
        'CustomerID': 'FRANK',
        'OrderDate': '1996-12-27T00:00:00.000Z',
        'ShippedDate': '1997-01-06T00:00:00.000Z',

```

```

    'Freight': 135.35,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10397,
    'CustomerID': 'PRINI',
    'OrderDate': '1996-12-27T00:00:00.000Z',
    'ShippedDate': '1997-01-02T00:00:00.000Z',
    'Freight': 60.26,
    'ShipName': 'Princesa Isabel Vinhos',
    'ShipAddress': 'Estrada da saúde n. 58',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,
    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 10398,
    'CustomerID': 'SAVEA',
    'OrderDate': '1996-12-30T00:00:00.000Z',
    'ShippedDate': '1997-01-09T00:00:00.000Z',
    'Freight': 89.16,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10399,
    'CustomerID': 'VAFFE',
    'OrderDate': '1996-12-31T00:00:00.000Z',
    'ShippedDate': '1997-01-08T00:00:00.000Z',
    'Freight': 27.36,
    'ShipName': 'Vaffeljernet',
    'ShipAddress': 'Smagsloget 45',
    'ShipCity': 'Århus',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10400,
    'CustomerID': 'EASTC',
    'OrderDate': '1997-01-01T00:00:00.000Z',
    'ShippedDate': '1997-01-16T00:00:00.000Z',
    'Freight': 83.93,
    'ShipName': 'Eastern Connection',
    'ShipAddress': '35 King George',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10401,

```

```

        'CustomerID': 'RATTC',
        'OrderDate': '1997-01-01T00:00:00.000Z',
        'ShippedDate': '1997-01-10T00:00:00.000Z',
        'Freight': 12.51,
        'ShipName': 'Rattlesnake Canyon Grocery',
        'ShipAddress': '2817 Milton Dr.',
        'ShipCity': 'Albuquerque',
        'ShipRegion': 'NM',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10402,
        'CustomerID': 'ERNSH',
        'OrderDate': '1997-01-02T00:00:00.000Z',
        'ShippedDate': '1997-01-10T00:00:00.000Z',
        'Freight': 67.88,
        'ShipName': 'Ernst Handel',
        'ShipAddress': 'Kirchgasse 6',
        'ShipCity': 'Graz',
        'ShipRegion': null,
        'ShipCountry': 'Austria'
    },
    {
        'OrderID': 10403,
        'CustomerID': 'ERNSH',
        'OrderDate': '1997-01-03T00:00:00.000Z',
        'ShippedDate': '1997-01-09T00:00:00.000Z',
        'Freight': 73.79,
        'ShipName': 'Ernst Handel',
        'ShipAddress': 'Kirchgasse 6',
        'ShipCity': 'Graz',
        'ShipRegion': null,
        'ShipCountry': 'Austria'
    },
    {
        'OrderID': 10404,
        'CustomerID': 'MAGAA',
        'OrderDate': '1997-01-03T00:00:00.000Z',
        'ShippedDate': '1997-01-08T00:00:00.000Z',
        'Freight': 155.97,
        'ShipName': 'Magazzini Alimentari Riuniti',
        'ShipAddress': 'Via Ludovico il Moro 22',
        'ShipCity': 'Bergamo',
        'ShipRegion': null,
        'ShipCountry': 'Italy'
    },
    {
        'OrderID': 10405,
        'CustomerID': 'LINOD',
        'OrderDate': '1997-01-06T00:00:00.000Z',
        'ShippedDate': '1997-01-22T00:00:00.000Z',
        'Freight': 34.82,
        'ShipName': 'LINO-Delicateses',
        'ShipAddress': 'Ave. 5 de Mayo Porlamar',
        'ShipCity': 'I. de Margarita',
        'ShipRegion': 'Nueva Esparta',
        'ShipCountry': 'Venezuela'
    }

```

```

},
{
  'OrderID': 10406,
  'CustomerID': 'QUEEN',
  'OrderDate': '1997-01-07T00:00:00.000Z',
  'ShippedDate': '1997-01-13T00:00:00.000Z',
  'Freight': 108.04,
  'ShipName': 'Queen Cozinha',
  'ShipAddress': 'Alameda dos Canários, 891',
  'ShipCity': 'Sao Paulo',
  'ShipRegion': 'SP',
  'ShipCountry': 'Brazil'
},
{
  'OrderID': 10407,
  'CustomerID': 'OTTIK',
  'OrderDate': '1997-01-07T00:00:00.000Z',
  'ShippedDate': '1997-01-30T00:00:00.000Z',
  'Freight': 91.48,
  'ShipName': 'Ottilies Käseladen',
  'ShipAddress': 'Mehrheimerstr. 369',
  'ShipCity': 'Köln',
  'ShipRegion': null,
  'ShipCountry': 'Germany'
},
{
  'OrderID': 10408,
  'CustomerID': 'FOLIG',
  'OrderDate': '1997-01-08T00:00:00.000Z',
  'ShippedDate': '1997-01-14T00:00:00.000Z',
  'Freight': 11.26,
  'ShipName': 'Folies gourmandes',
  'ShipAddress': '184, chaussée de Tournai',
  'ShipCity': 'Lille',
  'ShipRegion': null,
  'ShipCountry': 'France'
},
{
  'OrderID': 10409,
  'CustomerID': 'OCEAN',
  'OrderDate': '1997-01-09T00:00:00.000Z',
  'ShippedDate': '1997-01-14T00:00:00.000Z',
  'Freight': 29.83,
  'ShipName': 'Océano Atlántico Ltda.',
  'ShipAddress': 'Ing. Gustavo Moncada 8585 Piso 20-A',
  'ShipCity': 'Buenos Aires',
  'ShipRegion': null,
  'ShipCountry': 'Argentina'
},
{
  'OrderID': 10410,
  'CustomerID': 'BOTTM',
  'OrderDate': '1997-01-10T00:00:00.000Z',
  'ShippedDate': '1997-01-15T00:00:00.000Z',
  'Freight': 2.4,
  'ShipName': 'Bottom-Dollar Markets',
  'ShipAddress': '23 Tsawassen Blvd.',

```

```

        'ShipCity': 'Tsawassen',
        'ShipRegion': 'BC',
        'ShipCountry': 'Canada'
    },
    {
        'OrderID': 10411,
        'CustomerID': 'BOTTM',
        'OrderDate': '1997-01-10T00:00:00.000Z',
        'ShippedDate': '1997-01-21T00:00:00.000Z',
        'Freight': 23.65,
        'ShipName': 'Bottom-Dollar Markets',
        'ShipAddress': '23 Tsawassen Blvd.',
        'ShipCity': 'Tsawassen',
        'ShipRegion': 'BC',
        'ShipCountry': 'Canada'
    },
    {
        'OrderID': 10412,
        'CustomerID': 'WARTH',
        'OrderDate': '1997-01-13T00:00:00.000Z',
        'ShippedDate': '1997-01-15T00:00:00.000Z',
        'Freight': 3.77,
        'ShipName': 'Wartian Herkku',
        'ShipAddress': 'Torikatu 38',
        'ShipCity': 'Oulu',
        'ShipRegion': null,
        'ShipCountry': 'Finland'
    },
    {
        'OrderID': 10413,
        'CustomerID': 'LAMAI',
        'OrderDate': '1997-01-14T00:00:00.000Z',
        'ShippedDate': '1997-01-16T00:00:00.000Z',
        'Freight': 95.66,
        'ShipName': 'La maison d\'Asie',
        'ShipAddress': '1 rue Alsace-Lorraine',
        'ShipCity': 'Toulouse',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10414,
        'CustomerID': 'FAMIA',
        'OrderDate': '1997-01-14T00:00:00.000Z',
        'ShippedDate': '1997-01-17T00:00:00.000Z',
        'Freight': 21.48,
        'ShipName': 'Familia Arquibaldo',
        'ShipAddress': 'Rua Orós, 92',
        'ShipCity': 'Sao Paulo',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10415,
        'CustomerID': 'HUNGC',
        'OrderDate': '1997-01-15T00:00:00.000Z',
        'ShippedDate': '1997-01-24T00:00:00.000Z',

```



```

    'Freight': 0.2,
    'ShipName': 'Hungry Coyote Import Store',
    'ShipAddress': 'City Center Plaza 516 Main St.',
    'ShipCity': 'Elgin',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10416,
    'CustomerID': 'WARTH',
    'OrderDate': '1997-01-16T00:00:00.000Z',
    'ShippedDate': '1997-01-27T00:00:00.000Z',
    'Freight': 22.72,
    'ShipName': 'Wartian Herkku',
    'ShipAddress': 'Torikatu 38',
    'ShipCity': 'Oulu',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10417,
    'CustomerID': 'SIMOB',
    'OrderDate': '1997-01-16T00:00:00.000Z',
    'ShippedDate': '1997-01-28T00:00:00.000Z',
    'Freight': 70.29,
    'ShipName': 'Simons bistro',
    'ShipAddress': 'Vinbæltet 34',
    'ShipCity': 'Kobenhavn',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10418,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-01-17T00:00:00.000Z',
    'ShippedDate': '1997-01-24T00:00:00.000Z',
    'Freight': 17.55,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10419,
    'CustomerID': 'RICSU',
    'OrderDate': '1997-01-20T00:00:00.000Z',
    'ShippedDate': '1997-01-30T00:00:00.000Z',
    'Freight': 137.35,
    'ShipName': 'Richter Supermarkt',
    'ShipAddress': 'Starenweg 5',
    'ShipCity': 'Genève',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10420,

```

```

    'CustomerID': 'WELLI',
    'OrderDate': '1997-01-21T00:00:00.000Z',
    'ShippedDate': '1997-01-27T00:00:00.000Z',
    'Freight': 44.12,
    'ShipName': 'Wellington Importadora',
    'ShipAddress': 'Rua do Mercado, 12',
    'ShipCity': 'Resende',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10421,
    'CustomerID': 'QUEDE',
    'OrderDate': '1997-01-21T00:00:00.000Z',
    'ShippedDate': '1997-01-27T00:00:00.000Z',
    'Freight': 99.23,
    'ShipName': 'Que Delícia',
    'ShipAddress': 'Rua da Panificadora, 12',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10422,
    'CustomerID': 'FRANS',
    'OrderDate': '1997-01-22T00:00:00.000Z',
    'ShippedDate': '1997-01-31T00:00:00.000Z',
    'Freight': 3.02,
    'ShipName': 'Franchi S.p.A.',
    'ShipAddress': 'Via Monte Bianco 34',
    'ShipCity': 'Torino',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10423,
    'CustomerID': 'GOURL',
    'OrderDate': '1997-01-23T00:00:00.000Z',
    'ShippedDate': '1997-02-24T00:00:00.000Z',
    'Freight': 24.5,
    'ShipName': 'Gourmet Lanchonetes',
    'ShipAddress': 'Av. Brasil, 442',
    'ShipCity': 'Campinas',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10424,
    'CustomerID': 'MEREPE',
    'OrderDate': '1997-01-23T00:00:00.000Z',
    'ShippedDate': '1997-01-27T00:00:00.000Z',
    'Freight': 370.61,
    'ShipName': 'Mère Paillarde',
    'ShipAddress': '43 rue St. Laurent',
    'ShipCity': 'Montréal',
    'ShipRegion': 'Québec',
    'ShipCountry': 'Canada'
  }

```

```

},
{
  'OrderID': 10425,
  'CustomerID': 'LAMAI',
  'OrderDate': '1997-01-24T00:00:00.000Z',
  'ShippedDate': '1997-02-14T00:00:00.000Z',
  'Freight': 7.93,
  'ShipName': 'La maison d\'Asie',
  'ShipAddress': '1 rue Alsace-Lorraine',
  'ShipCity': 'Toulouse',
  'ShipRegion': null,
  'ShipCountry': 'France'
},
{
  'OrderID': 10426,
  'CustomerID': 'GALED',
  'OrderDate': '1997-01-27T00:00:00.000Z',
  'ShippedDate': '1997-02-06T00:00:00.000Z',
  'Freight': 18.69,
  'ShipName': 'Galería del gastronómo',
  'ShipAddress': 'Rambla de Cataluña, 23',
  'ShipCity': 'Barcelona',
  'ShipRegion': null,
  'ShipCountry': 'Spain'
},
{
  'OrderID': 10427,
  'CustomerID': 'PICCO',
  'OrderDate': '1997-01-27T00:00:00.000Z',
  'ShippedDate': '1997-03-03T00:00:00.000Z',
  'Freight': 31.29,
  'ShipName': 'Piccolo und mehr',
  'ShipAddress': 'Geislweg 14',
  'ShipCity': 'Salzburg',
  'ShipRegion': null,
  'ShipCountry': 'Austria'
},
{
  'OrderID': 10428,
  'CustomerID': 'REGGC',
  'OrderDate': '1997-01-28T00:00:00.000Z',
  'ShippedDate': '1997-02-04T00:00:00.000Z',
  'Freight': 11.09,
  'ShipName': 'Reggiani Caseifici',
  'ShipAddress': 'Strada Provinciale 124',
  'ShipCity': 'Reggio Emilia',
  'ShipRegion': null,
  'ShipCountry': 'Italy'
},
{
  'OrderID': 10429,
  'CustomerID': 'HUNGO',
  'OrderDate': '1997-01-29T00:00:00.000Z',
  'ShippedDate': '1997-02-07T00:00:00.000Z',
  'Freight': 56.63,
  'ShipName': 'Hungry Owl All-Night Grocers',
  'ShipAddress': '8 Johnstown Road',

```

```

    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10430,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-01-30T00:00:00.000Z',
    'ShippedDate': '1997-02-03T00:00:00.000Z',
    'Freight': 458.78,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10431,
    'CustomerID': 'BOTTM',
    'OrderDate': '1997-01-30T00:00:00.000Z',
    'ShippedDate': '1997-02-07T00:00:00.000Z',
    'Freight': 44.17,
    'ShipName': 'Bottom-Dollar Markets',
    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10432,
    'CustomerID': 'SPLIR',
    'OrderDate': '1997-01-31T00:00:00.000Z',
    'ShippedDate': '1997-02-07T00:00:00.000Z',
    'Freight': 4.34,
    'ShipName': 'Split Rail Beer & Ale',
    'ShipAddress': 'P.O. Box 555',
    'ShipCity': 'Lander',
    'ShipRegion': 'WY',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10433,
    'CustomerID': 'PRINI',
    'OrderDate': '1997-02-03T00:00:00.000Z',
    'ShippedDate': '1997-03-04T00:00:00.000Z',
    'Freight': 73.83,
    'ShipName': 'Princesa Isabel Vinhos',
    'ShipAddress': 'Estrada da saúde n. 58',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,
    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 10434,
    'CustomerID': 'FOLKO',
    'OrderDate': '1997-02-03T00:00:00.000Z',
    'ShippedDate': '1997-02-13T00:00:00.000Z',

```

```

    'Freight': 17.92,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10435,
    'CustomerID': 'CONSH',
    'OrderDate': '1997-02-04T00:00:00.000Z',
    'ShippedDate': '1997-02-07T00:00:00.000Z',
    'Freight': 9.21,
    'ShipName': 'Consolidated Holdings',
    'ShipAddress': 'Berkeley Gardens 12 Brewery',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10436,
    'CustomerID': 'BLONP',
    'OrderDate': '1997-02-05T00:00:00.000Z',
    'ShippedDate': '1997-02-11T00:00:00.000Z',
    'Freight': 156.66,
    'ShipName': 'Blondel père et fils',
    'ShipAddress': '24, place Kléber',
    'ShipCity': 'Strasbourg',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10437,
    'CustomerID': 'WARTH',
    'OrderDate': '1997-02-05T00:00:00.000Z',
    'ShippedDate': '1997-02-12T00:00:00.000Z',
    'Freight': 19.97,
    'ShipName': 'Wartian Herkku',
    'ShipAddress': 'Torikatu 38',
    'ShipCity': 'Oulu',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10438,
    'CustomerID': 'TOMSP',
    'OrderDate': '1997-02-06T00:00:00.000Z',
    'ShippedDate': '1997-02-14T00:00:00.000Z',
    'Freight': 8.24,
    'ShipName': 'Toms Spezialitäten',
    'ShipAddress': 'Luisenstr. 48',
    'ShipCity': 'Münster',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10439,

```

```

        'CustomerID': 'MEREP',
        'OrderDate': '1997-02-07T00:00:00.000Z',
        'ShippedDate': '1997-02-10T00:00:00.000Z',
        'Freight': 4.07,
        'ShipName': 'Mère Paillarde',
        'ShipAddress': '43 rue St. Laurent',
        'ShipCity': 'Montréal',
        'ShipRegion': 'Québec',
        'ShipCountry': 'Canada'
    },
    {
        'OrderID': 10440,
        'CustomerID': 'SAVEA',
        'OrderDate': '1997-02-10T00:00:00.000Z',
        'ShippedDate': '1997-02-28T00:00:00.000Z',
        'Freight': 86.53,
        'ShipName': 'Save-a-lot Markets',
        'ShipAddress': '187 Suffolk Ln.',
        'ShipCity': 'Boise',
        'ShipRegion': 'ID',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10441,
        'CustomerID': 'OLDWO',
        'OrderDate': '1997-02-10T00:00:00.000Z',
        'ShippedDate': '1997-03-14T00:00:00.000Z',
        'Freight': 73.02,
        'ShipName': 'Old World Delicatessen',
        'ShipAddress': '2743 Bering St.',
        'ShipCity': 'Anchorage',
        'ShipRegion': 'AK',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10442,
        'CustomerID': 'ERNSH',
        'OrderDate': '1997-02-11T00:00:00.000Z',
        'ShippedDate': '1997-02-18T00:00:00.000Z',
        'Freight': 47.94,
        'ShipName': 'Ernst Handel',
        'ShipAddress': 'Kirchgasse 6',
        'ShipCity': 'Graz',
        'ShipRegion': null,
        'ShipCountry': 'Austria'
    },
    {
        'OrderID': 10443,
        'CustomerID': 'REGGC',
        'OrderDate': '1997-02-12T00:00:00.000Z',
        'ShippedDate': '1997-02-14T00:00:00.000Z',
        'Freight': 13.95,
        'ShipName': 'Reggiani Caseifici',
        'ShipAddress': 'Strada Provinciale 124',
        'ShipCity': 'Reggio Emilia',
        'ShipRegion': null,
        'ShipCountry': 'Italy'
    }

```

```

    },
    {
        'OrderID': 10444,
        'CustomerID': 'BERGS',
        'OrderDate': '1997-02-12T00:00:00.000Z',
        'ShippedDate': '1997-02-21T00:00:00.000Z',
        'Freight': 3.5,
        'ShipName': 'Berglunds snabbköp',
        'ShipAddress': 'Berguvsvägen 8',
        'ShipCity': 'Luleå',
        'ShipRegion': null,
        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 10445,
        'CustomerID': 'BERGS',
        'OrderDate': '1997-02-13T00:00:00.000Z',
        'ShippedDate': '1997-02-20T00:00:00.000Z',
        'Freight': 9.3,
        'ShipName': 'Berglunds snabbköp',
        'ShipAddress': 'Berguvsvägen 8',
        'ShipCity': 'Luleå',
        'ShipRegion': null,
        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 10446,
        'CustomerID': 'TOMSP',
        'OrderDate': '1997-02-14T00:00:00.000Z',
        'ShippedDate': '1997-02-19T00:00:00.000Z',
        'Freight': 14.68,
        'ShipName': 'Toms Spezialitäten',
        'ShipAddress': 'Luisenstr. 48',
        'ShipCity': 'Münster',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10447,
        'CustomerID': 'RICAR',
        'OrderDate': '1997-02-14T00:00:00.000Z',
        'ShippedDate': '1997-03-07T00:00:00.000Z',
        'Freight': 68.66,
        'ShipName': 'Ricardo Adocicados',
        'ShipAddress': 'Av. Copacabana, 267',
        'ShipCity': 'Rio de Janeiro',
        'ShipRegion': 'RJ',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10448,
        'CustomerID': 'RANCH',
        'OrderDate': '1997-02-17T00:00:00.000Z',
        'ShippedDate': '1997-02-24T00:00:00.000Z',
        'Freight': 38.82,
        'ShipName': 'Rancho grande',
        'ShipAddress': 'Av. del Libertador 900',
    }

```

```

    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 10449,
    'CustomerID': 'BLONP',
    'OrderDate': '1997-02-18T00:00:00.000Z',
    'ShippedDate': '1997-02-27T00:00:00.000Z',
    'Freight': 53.3,
    'ShipName': 'Blondel père et fils',
    'ShipAddress': '24, place Kléber',
    'ShipCity': 'Strasbourg',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10450,
    'CustomerID': 'VICTE',
    'OrderDate': '1997-02-19T00:00:00.000Z',
    'ShippedDate': '1997-03-11T00:00:00.000Z',
    'Freight': 7.23,
    'ShipName': 'Victuailles en stock',
    'ShipAddress': '2, rue du Commerce',
    'ShipCity': 'Lyon',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10451,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-02-19T00:00:00.000Z',
    'ShippedDate': '1997-03-12T00:00:00.000Z',
    'Freight': 189.09,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10452,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-02-20T00:00:00.000Z',
    'ShippedDate': '1997-02-26T00:00:00.000Z',
    'Freight': 140.26,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10453,
    'CustomerID': 'AROUT',
    'OrderDate': '1997-02-21T00:00:00.000Z',
    'ShippedDate': '1997-02-26T00:00:00.000Z',

```



```

    'Freight': 25.36,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10454,
    'CustomerID': 'LAMAI',
    'OrderDate': '1997-02-21T00:00:00.000Z',
    'ShippedDate': '1997-02-25T00:00:00.000Z',
    'Freight': 2.74,
    'ShipName': 'La maison d\'Asie',
    'ShipAddress': '1 rue Alsace-Lorraine',
    'ShipCity': 'Toulouse',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10455,
    'CustomerID': 'WARTH',
    'OrderDate': '1997-02-24T00:00:00.000Z',
    'ShippedDate': '1997-03-03T00:00:00.000Z',
    'Freight': 180.45,
    'ShipName': 'Wartian Herkku',
    'ShipAddress': 'Torikatu 38',
    'ShipCity': 'Oulu',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10456,
    'CustomerID': 'KOENE',
    'OrderDate': '1997-02-25T00:00:00.000Z',
    'ShippedDate': '1997-02-28T00:00:00.000Z',
    'Freight': 8.12,
    'ShipName': 'Königlich Essen',
    'ShipAddress': 'Maubelstr. 90',
    'ShipCity': 'Brandenburg',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10457,
    'CustomerID': 'KOENE',
    'OrderDate': '1997-02-25T00:00:00.000Z',
    'ShippedDate': '1997-03-03T00:00:00.000Z',
    'Freight': 11.57,
    'ShipName': 'Königlich Essen',
    'ShipAddress': 'Maubelstr. 90',
    'ShipCity': 'Brandenburg',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10458,

```

```

    'CustomerID': 'SUPRD',
    'OrderDate': '1997-02-26T00:00:00.000Z',
    'ShippedDate': '1997-03-04T00:00:00.000Z',
    'Freight': 147.06,
    'ShipName': 'Suprêmes délices',
    'ShipAddress': 'Boulevard Tirou, 255',
    'ShipCity': 'Charleroi',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10459,
    'CustomerID': 'VICTE',
    'OrderDate': '1997-02-27T00:00:00.000Z',
    'ShippedDate': '1997-02-28T00:00:00.000Z',
    'Freight': 25.09,
    'ShipName': 'Victuailles en stock',
    'ShipAddress': '2, rue du Commerce',
    'ShipCity': 'Lyon',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10460,
    'CustomerID': 'FOLKO',
    'OrderDate': '1997-02-28T00:00:00.000Z',
    'ShippedDate': '1997-03-03T00:00:00.000Z',
    'Freight': 16.27,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10461,
    'CustomerID': 'LILAS',
    'OrderDate': '1997-02-28T00:00:00.000Z',
    'ShippedDate': '1997-03-05T00:00:00.000Z',
    'Freight': 148.61,
    'ShipName': 'LILA-Supermercado',
    'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10462,
    'CustomerID': 'CONSH',
    'OrderDate': '1997-03-03T00:00:00.000Z',
    'ShippedDate': '1997-03-18T00:00:00.000Z',
    'Freight': 6.17,
    'ShipName': 'Consolidated Holdings',
    'ShipAddress': 'Berkeley Gardens 12 Brewery',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  }

```

```

    },
    {
        'OrderID': 10463,
        'CustomerID': 'SUPRD',
        'OrderDate': '1997-03-04T00:00:00.000Z',
        'ShippedDate': '1997-03-06T00:00:00.000Z',
        'Freight': 14.78,
        'ShipName': 'Suprêmes délices',
        'ShipAddress': 'Boulevard Tirou, 255',
        'ShipCity': 'Charleroi',
        'ShipRegion': null,
        'ShipCountry': 'Belgium'
    },
    {
        'OrderID': 10464,
        'CustomerID': 'FURIB',
        'OrderDate': '1997-03-04T00:00:00.000Z',
        'ShippedDate': '1997-03-14T00:00:00.000Z',
        'Freight': 89,
        'ShipName': 'Furia Bacalhau e Frutos do Mar',
        'ShipAddress': 'Jardim das rosas n. 32',
        'ShipCity': 'Lisboa',
        'ShipRegion': null,
        'ShipCountry': 'Portugal'
    },
    {
        'OrderID': 10465,
        'CustomerID': 'VAFFE',
        'OrderDate': '1997-03-05T00:00:00.000Z',
        'ShippedDate': '1997-03-14T00:00:00.000Z',
        'Freight': 145.04,
        'ShipName': 'Vaffeljernet',
        'ShipAddress': 'Smagsloget 45',
        'ShipCity': 'Århus',
        'ShipRegion': null,
        'ShipCountry': 'Denmark'
    },
    {
        'OrderID': 10466,
        'CustomerID': 'COMMI',
        'OrderDate': '1997-03-06T00:00:00.000Z',
        'ShippedDate': '1997-03-13T00:00:00.000Z',
        'Freight': 11.93,
        'ShipName': 'Comércio Mineiro',
        'ShipAddress': 'Av. dos Lusíadas, 23',
        'ShipCity': 'Sao Paulo',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10467,
        'CustomerID': 'MAGAA',
        'OrderDate': '1997-03-06T00:00:00.000Z',
        'ShippedDate': '1997-03-11T00:00:00.000Z',
        'Freight': 4.93,
        'ShipName': 'Magazzini Alimentari Riuniti',
        'ShipAddress': 'Via Ludovico il Moro 22',
    }

```

```

    'ShipCity': 'Bergamo',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10468,
    'CustomerID': 'KOENE',
    'OrderDate': '1997-03-07T00:00:00.000Z',
    'ShippedDate': '1997-03-12T00:00:00.000Z',
    'Freight': 44.12,
    'ShipName': 'Königlich Essen',
    'ShipAddress': 'Maubelstr. 90',
    'ShipCity': 'Brandenburg',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10469,
    'CustomerID': 'WHITC',
    'OrderDate': '1997-03-10T00:00:00.000Z',
    'ShippedDate': '1997-03-14T00:00:00.000Z',
    'Freight': 60.18,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10470,
    'CustomerID': 'BONAP',
    'OrderDate': '1997-03-11T00:00:00.000Z',
    'ShippedDate': '1997-03-14T00:00:00.000Z',
    'Freight': 64.56,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10471,
    'CustomerID': 'BSBEV',
    'OrderDate': '1997-03-11T00:00:00.000Z',
    'ShippedDate': '1997-03-18T00:00:00.000Z',
    'Freight': 45.59,
    'ShipName': 'B\' Beverages',
    'ShipAddress': 'Fauntleroy Circus',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10472,
    'CustomerID': 'SEVES',
    'OrderDate': '1997-03-12T00:00:00.000Z',
    'ShippedDate': '1997-03-19T00:00:00.000Z',

```

```

    'Freight': 4.2,
    'ShipName': 'Seven Seas Imports',
    'ShipAddress': '90 Wadhurst Rd.',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10473,
    'CustomerID': 'ISLAT',
    'OrderDate': '1997-03-13T00:00:00.000Z',
    'ShippedDate': '1997-03-21T00:00:00.000Z',
    'Freight': 16.37,
    'ShipName': 'Island Trading',
    'ShipAddress': 'Garden House Crowther Way',
    'ShipCity': 'Cowes',
    'ShipRegion': 'Isle of Wight',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10474,
    'CustomerID': 'PERIC',
    'OrderDate': '1997-03-13T00:00:00.000Z',
    'ShippedDate': '1997-03-21T00:00:00.000Z',
    'Freight': 83.49,
    'ShipName': 'Pericles Comidas clásicas',
    'ShipAddress': 'Calle Dr. Jorge Cash 321',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10475,
    'CustomerID': 'SUPRD',
    'OrderDate': '1997-03-14T00:00:00.000Z',
    'ShippedDate': '1997-04-04T00:00:00.000Z',
    'Freight': 68.52,
    'ShipName': 'Suprêmes délices',
    'ShipAddress': 'Boulevard Tirou, 255',
    'ShipCity': 'Charleroi',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10476,
    'CustomerID': 'HILAA',
    'OrderDate': '1997-03-17T00:00:00.000Z',
    'ShippedDate': '1997-03-24T00:00:00.000Z',
    'Freight': 4.41,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10477,

```

```

    'CustomerID': 'PRINI',
    'OrderDate': '1997-03-17T00:00:00.000Z',
    'ShippedDate': '1997-03-25T00:00:00.000Z',
    'Freight': 13.02,
    'ShipName': 'Princesa Isabel Vinhos',
    'ShipAddress': 'Estrada da saúde n. 58',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,
    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 10478,
    'CustomerID': 'VICTE',
    'OrderDate': '1997-03-18T00:00:00.000Z',
    'ShippedDate': '1997-03-26T00:00:00.000Z',
    'Freight': 4.81,
    'ShipName': 'Victuailles en stock',
    'ShipAddress': '2, rue du Commerce',
    'ShipCity': 'Lyon',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10479,
    'CustomerID': 'RATTC',
    'OrderDate': '1997-03-19T00:00:00.000Z',
    'ShippedDate': '1997-03-21T00:00:00.000Z',
    'Freight': 708.95,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10480,
    'CustomerID': 'FOLIG',
    'OrderDate': '1997-03-20T00:00:00.000Z',
    'ShippedDate': '1997-03-24T00:00:00.000Z',
    'Freight': 1.35,
    'ShipName': 'Folies gourmandes',
    'ShipAddress': '184, chaussée de Tournai',
    'ShipCity': 'Lille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10481,
    'CustomerID': 'RICAR',
    'OrderDate': '1997-03-20T00:00:00.000Z',
    'ShippedDate': '1997-03-25T00:00:00.000Z',
    'Freight': 64.33,
    'ShipName': 'Ricardo Adocicados',
    'ShipAddress': 'Av. Copacabana, 267',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  }

```

```

    },
    {
        'OrderID': 10482,
        'CustomerID': 'LAZYK',
        'OrderDate': '1997-03-21T00:00:00.000Z',
        'ShippedDate': '1997-04-10T00:00:00.000Z',
        'Freight': 7.48,
        'ShipName': 'Lazy K Kountry Store',
        'ShipAddress': '12 Orchestra Terrace',
        'ShipCity': 'Walla Walla',
        'ShipRegion': 'WA',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10483,
        'CustomerID': 'WHITC',
        'OrderDate': '1997-03-24T00:00:00.000Z',
        'ShippedDate': '1997-04-25T00:00:00.000Z',
        'Freight': 15.28,
        'ShipName': 'White Clover Markets',
        'ShipAddress': '1029 - 12th Ave. S.',
        'ShipCity': 'Seattle',
        'ShipRegion': 'WA',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10484,
        'CustomerID': 'BSBEV',
        'OrderDate': '1997-03-24T00:00:00.000Z',
        'ShippedDate': '1997-04-01T00:00:00.000Z',
        'Freight': 6.88,
        'ShipName': 'B\ Beverages',
        'ShipAddress': 'Fauntleroy Circus',
        'ShipCity': 'London',
        'ShipRegion': null,
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10485,
        'CustomerID': 'LINOD',
        'OrderDate': '1997-03-25T00:00:00.000Z',
        'ShippedDate': '1997-03-31T00:00:00.000Z',
        'Freight': 64.45,
        'ShipName': 'LINO-Delicatesses',
        'ShipAddress': 'Ave. 5 de Mayo Porlamar',
        'ShipCity': 'I. de Margarita',
        'ShipRegion': 'Nueva Esparta',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10486,
        'CustomerID': 'HILAA',
        'OrderDate': '1997-03-26T00:00:00.000Z',
        'ShippedDate': '1997-04-02T00:00:00.000Z',
        'Freight': 30.53,
        'ShipName': 'HILARION-Abastos',
        'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    }

```

```

    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10487,
    'CustomerID': 'QUEEN',
    'OrderDate': '1997-03-26T00:00:00.000Z',
    'ShippedDate': '1997-03-28T00:00:00.000Z',
    'Freight': 71.07,
    'ShipName': 'Queen Cozinha',
    'ShipAddress': 'Alameda dos Canários, 891',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10488,
    'CustomerID': 'FRANK',
    'OrderDate': '1997-03-27T00:00:00.000Z',
    'ShippedDate': '1997-04-02T00:00:00.000Z',
    'Freight': 4.93,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10489,
    'CustomerID': 'PICCO',
    'OrderDate': '1997-03-28T00:00:00.000Z',
    'ShippedDate': '1997-04-09T00:00:00.000Z',
    'Freight': 5.29,
    'ShipName': 'Piccolo und mehr',
    'ShipAddress': 'Geislweg 14',
    'ShipCity': 'Salzburg',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10490,
    'CustomerID': 'HILAA',
    'OrderDate': '1997-03-31T00:00:00.000Z',
    'ShippedDate': '1997-04-03T00:00:00.000Z',
    'Freight': 210.19,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10491,
    'CustomerID': 'FURIB',
    'OrderDate': '1997-03-31T00:00:00.000Z',
    'ShippedDate': '1997-04-08T00:00:00.000Z',

```



```

    'Freight': 16.96,
    'ShipName': 'Furia Bacalhau e Frutos do Mar',
    'ShipAddress': 'Jardim das rosas n. 32',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,
    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 10492,
    'CustomerID': 'BOTTM',
    'OrderDate': '1997-04-01T00:00:00.000Z',
    'ShippedDate': '1997-04-11T00:00:00.000Z',
    'Freight': 62.89,
    'ShipName': 'Bottom-Dollar Markets',
    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10493,
    'CustomerID': 'LAMAI',
    'OrderDate': '1997-04-02T00:00:00.000Z',
    'ShippedDate': '1997-04-10T00:00:00.000Z',
    'Freight': 10.64,
    'ShipName': 'La maison d\'Asie',
    'ShipAddress': '1 rue Alsace-Lorraine',
    'ShipCity': 'Toulouse',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10494,
    'CustomerID': 'COMMI',
    'OrderDate': '1997-04-02T00:00:00.000Z',
    'ShippedDate': '1997-04-09T00:00:00.000Z',
    'Freight': 65.99,
    'ShipName': 'Comércio Mineiro',
    'ShipAddress': 'Av. dos Lusíadas, 23',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10495,
    'CustomerID': 'LAUGB',
    'OrderDate': '1997-04-03T00:00:00.000Z',
    'ShippedDate': '1997-04-11T00:00:00.000Z',
    'Freight': 4.65,
    'ShipName': 'Laughing Bacchus Wine Cellars',
    'ShipAddress': '2319 Elm St.',
    'ShipCity': 'Vancouver',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10496,

```

```

    'CustomerID': 'TRADH',
    'OrderDate': '1997-04-04T00:00:00.000Z',
    'ShippedDate': '1997-04-07T00:00:00.000Z',
    'Freight': 46.77,
    'ShipName': 'Tradiçao Hipermercados',
    'ShipAddress': 'Av. Inês de Castro, 414',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10497,
    'CustomerID': 'LEHMS',
    'OrderDate': '1997-04-04T00:00:00.000Z',
    'ShippedDate': '1997-04-07T00:00:00.000Z',
    'Freight': 36.21,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10498,
    'CustomerID': 'HILAA',
    'OrderDate': '1997-04-07T00:00:00.000Z',
    'ShippedDate': '1997-04-11T00:00:00.000Z',
    'Freight': 29.75,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10499,
    'CustomerID': 'LILAS',
    'OrderDate': '1997-04-08T00:00:00.000Z',
    'ShippedDate': '1997-04-16T00:00:00.000Z',
    'Freight': 102.02,
    'ShipName': 'LILA-Supermercado',
    'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10500,
    'CustomerID': 'LAMAI',
    'OrderDate': '1997-04-09T00:00:00.000Z',
    'ShippedDate': '1997-04-17T00:00:00.000Z',
    'Freight': 42.68,
    'ShipName': 'La maison d\'Asie',
    'ShipAddress': '1 rue Alsace-Lorraine',
    'ShipCity': 'Toulouse',
    'ShipRegion': null,
    'ShipCountry': 'France'
  }

```

```

    },
    {
        'OrderID': 10501,
        'CustomerID': 'BLAUS',
        'OrderDate': '1997-04-09T00:00:00.000Z',
        'ShippedDate': '1997-04-16T00:00:00.000Z',
        'Freight': 8.85,
        'ShipName': 'Blauer See Delikatessen',
        'ShipAddress': 'Forsterstr. 57',
        'ShipCity': 'Mannheim',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10502,
        'CustomerID': 'PERIC',
        'OrderDate': '1997-04-10T00:00:00.000Z',
        'ShippedDate': '1997-04-29T00:00:00.000Z',
        'Freight': 69.32,
        'ShipName': 'Pericles Comidas clásicas',
        'ShipAddress': 'Calle Dr. Jorge Cash 321',
        'ShipCity': 'México D.F.',
        'ShipRegion': null,
        'ShipCountry': 'Mexico'
    },
    {
        'OrderID': 10503,
        'CustomerID': 'HUNGO',
        'OrderDate': '1997-04-11T00:00:00.000Z',
        'ShippedDate': '1997-04-16T00:00:00.000Z',
        'Freight': 16.74,
        'ShipName': 'Hungry Owl All-Night Grocers',
        'ShipAddress': '8 Johnstown Road',
        'ShipCity': 'Cork',
        'ShipRegion': 'Co. Cork',
        'ShipCountry': 'Ireland'
    },
    {
        'OrderID': 10504,
        'CustomerID': 'WHITC',
        'OrderDate': '1997-04-11T00:00:00.000Z',
        'ShippedDate': '1997-04-18T00:00:00.000Z',
        'Freight': 59.13,
        'ShipName': 'White Clover Markets',
        'ShipAddress': '1029 - 12th Ave. S.',
        'ShipCity': 'Seattle',
        'ShipRegion': 'WA',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10505,
        'CustomerID': 'MEREPA',
        'OrderDate': '1997-04-14T00:00:00.000Z',
        'ShippedDate': '1997-04-21T00:00:00.000Z',
        'Freight': 7.13,
        'ShipName': 'Mère Paillarde',
        'ShipAddress': '43 rue St. Laurent',
    }

```

```

        'ShipCity': 'Montréal',
        'ShipRegion': 'Québec',
        'ShipCountry': 'Canada'
    },
    {
        'OrderID': 10506,
        'CustomerID': 'KOENE',
        'OrderDate': '1997-04-15T00:00:00.000Z',
        'ShippedDate': '1997-05-02T00:00:00.000Z',
        'Freight': 21.19,
        'ShipName': 'Königlich Essen',
        'ShipAddress': 'Maubelstr. 90',
        'ShipCity': 'Brandenburg',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10507,
        'CustomerID': 'ANTON',
        'OrderDate': '1997-04-15T00:00:00.000Z',
        'ShippedDate': '1997-04-22T00:00:00.000Z',
        'Freight': 47.45,
        'ShipName': 'Antonio Moreno Taquería',
        'ShipAddress': 'Mataderos 2312',
        'ShipCity': 'México D.F.',
        'ShipRegion': null,
        'ShipCountry': 'Mexico'
    },
    {
        'OrderID': 10508,
        'CustomerID': 'OTTIK',
        'OrderDate': '1997-04-16T00:00:00.000Z',
        'ShippedDate': '1997-05-13T00:00:00.000Z',
        'Freight': 4.99,
        'ShipName': 'Ottilies Käseladen',
        'ShipAddress': 'Mehrheimerstr. 369',
        'ShipCity': 'Köln',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10509,
        'CustomerID': 'BLAUS',
        'OrderDate': '1997-04-17T00:00:00.000Z',
        'ShippedDate': '1997-04-29T00:00:00.000Z',
        'Freight': 0.15,
        'ShipName': 'Blauer See Delikatessen',
        'ShipAddress': 'Forsterstr. 57',
        'ShipCity': 'Mannheim',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10510,
        'CustomerID': 'SAVEA',
        'OrderDate': '1997-04-18T00:00:00.000Z',
        'ShippedDate': '1997-04-28T00:00:00.000Z',

```

```

    'Freight': 367.63,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10511,
    'CustomerID': 'BONAP',
    'OrderDate': '1997-04-18T00:00:00.000Z',
    'ShippedDate': '1997-04-21T00:00:00.000Z',
    'Freight': 350.64,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10512,
    'CustomerID': 'FAMIA',
    'OrderDate': '1997-04-21T00:00:00.000Z',
    'ShippedDate': '1997-04-24T00:00:00.000Z',
    'Freight': 3.53,
    'ShipName': 'Familia Arquibaldo',
    'ShipAddress': 'Rua Orós, 92',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10513,
    'CustomerID': 'WANDK',
    'OrderDate': '1997-04-22T00:00:00.000Z',
    'ShippedDate': '1997-04-28T00:00:00.000Z',
    'Freight': 105.65,
    'ShipName': 'Die Wandernde Kuh',
    'ShipAddress': 'Adenauerallee 900',
    'ShipCity': 'Stuttgart',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10514,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-04-22T00:00:00.000Z',
    'ShippedDate': '1997-05-16T00:00:00.000Z',
    'Freight': 789.95,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10515,

```

```

    'CustomerID': 'QUICK',
    'OrderDate': '1997-04-23T00:00:00.000Z',
    'ShippedDate': '1997-05-23T00:00:00.000Z',
    'Freight': 204.47,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10516,
    'CustomerID': 'HUNGO',
    'OrderDate': '1997-04-24T00:00:00.000Z',
    'ShippedDate': '1997-05-01T00:00:00.000Z',
    'Freight': 62.78,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10517,
    'CustomerID': 'NORTS',
    'OrderDate': '1997-04-24T00:00:00.000Z',
    'ShippedDate': '1997-04-29T00:00:00.000Z',
    'Freight': 32.07,
    'ShipName': 'North/South',
    'ShipAddress': 'South House 300 Queensbridge',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10518,
    'CustomerID': 'TORTU',
    'OrderDate': '1997-04-25T00:00:00.000Z',
    'ShippedDate': '1997-05-05T00:00:00.000Z',
    'Freight': 218.15,
    'ShipName': 'Tortuga Restaurante',
    'ShipAddress': 'Avda. Azteca 123',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10519,
    'CustomerID': 'CHOPS',
    'OrderDate': '1997-04-28T00:00:00.000Z',
    'ShippedDate': '1997-05-01T00:00:00.000Z',
    'Freight': 91.76,
    'ShipName': 'Chop-suey Chinese',
    'ShipAddress': 'Hauptstr. 31',
    'ShipCity': 'Bern',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  }

```

```

    },
    {
        'OrderID': 10520,
        'CustomerID': 'SANTG',
        'OrderDate': '1997-04-29T00:00:00.000Z',
        'ShippedDate': '1997-05-01T00:00:00.000Z',
        'Freight': 13.37,
        'ShipName': 'Santé Gourmet',
        'ShipAddress': 'Erling Skakkes gate 78',
        'ShipCity': 'Stavern',
        'ShipRegion': null,
        'ShipCountry': 'Norway'
    },
    {
        'OrderID': 10521,
        'CustomerID': 'CACTU',
        'OrderDate': '1997-04-29T00:00:00.000Z',
        'ShippedDate': '1997-05-02T00:00:00.000Z',
        'Freight': 17.22,
        'ShipName': 'Cactus Comidas para llevar',
        'ShipAddress': 'Cerrito 333',
        'ShipCity': 'Buenos Aires',
        'ShipRegion': null,
        'ShipCountry': 'Argentina'
    },
    {
        'OrderID': 10522,
        'CustomerID': 'LEHMS',
        'OrderDate': '1997-04-30T00:00:00.000Z',
        'ShippedDate': '1997-05-06T00:00:00.000Z',
        'Freight': 45.33,
        'ShipName': 'Lehmanns Marktstand',
        'ShipAddress': 'Magazinweg 7',
        'ShipCity': 'Frankfurt a.M.',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10523,
        'CustomerID': 'SEVES',
        'OrderDate': '1997-05-01T00:00:00.000Z',
        'ShippedDate': '1997-05-30T00:00:00.000Z',
        'Freight': 77.63,
        'ShipName': 'Seven Seas Imports',
        'ShipAddress': '90 Wadhurst Rd.',
        'ShipCity': 'London',
        'ShipRegion': null,
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10524,
        'CustomerID': 'BERGS',
        'OrderDate': '1997-05-01T00:00:00.000Z',
        'ShippedDate': '1997-05-07T00:00:00.000Z',
        'Freight': 244.79,
        'ShipName': 'Berglunds snabbköp',
        'ShipAddress': 'Berguvsvägen 8',
    }

```

```

        'ShipCity': 'Luleå',
        'ShipRegion': null,
        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 10525,
        'CustomerID': 'BONAP',
        'OrderDate': '1997-05-02T00:00:00.000Z',
        'ShippedDate': '1997-05-23T00:00:00.000Z',
        'Freight': 11.06,
        'ShipName': 'Bon app',
        'ShipAddress': '12, rue des Bouchers',
        'ShipCity': 'Marseille',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10526,
        'CustomerID': 'WARTH',
        'OrderDate': '1997-05-05T00:00:00.000Z',
        'ShippedDate': '1997-05-15T00:00:00.000Z',
        'Freight': 58.59,
        'ShipName': 'Wartian Herkku',
        'ShipAddress': 'Torikatu 38',
        'ShipCity': 'Oulu',
        'ShipRegion': null,
        'ShipCountry': 'Finland'
    },
    {
        'OrderID': 10527,
        'CustomerID': 'QUICK',
        'OrderDate': '1997-05-05T00:00:00.000Z',
        'ShippedDate': '1997-05-07T00:00:00.000Z',
        'Freight': 41.9,
        'ShipName': 'QUICK-Stop',
        'ShipAddress': 'Taucherstraße 10',
        'ShipCity': 'Cunewalde',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10528,
        'CustomerID': 'GREAL',
        'OrderDate': '1997-05-06T00:00:00.000Z',
        'ShippedDate': '1997-05-09T00:00:00.000Z',
        'Freight': 3.35,
        'ShipName': 'Great Lakes Food Market',
        'ShipAddress': '2732 Baker Blvd.',
        'ShipCity': 'Eugene',
        'ShipRegion': 'OR',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10529,
        'CustomerID': 'MAISD',
        'OrderDate': '1997-05-07T00:00:00.000Z',
        'ShippedDate': '1997-05-09T00:00:00.000Z',

```



```

    'Freight': 66.69,
    'ShipName': 'Maison Dewey',
    'ShipAddress': 'Rue Joseph-Bens 532',
    'ShipCity': 'Bruxelles',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10530,
    'CustomerID': 'PICCO',
    'OrderDate': '1997-05-08T00:00:00.000Z',
    'ShippedDate': '1997-05-12T00:00:00.000Z',
    'Freight': 339.22,
    'ShipName': 'Piccolo und mehr',
    'ShipAddress': 'Geislweg 14',
    'ShipCity': 'Salzburg',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10531,
    'CustomerID': 'OCEAN',
    'OrderDate': '1997-05-08T00:00:00.000Z',
    'ShippedDate': '1997-05-19T00:00:00.000Z',
    'Freight': 8.12,
    'ShipName': 'Océano Atlántico Ltda.',
    'ShipAddress': 'Ing. Gustavo Moncada 8585 Piso 20-A',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 10532,
    'CustomerID': 'EASTC',
    'OrderDate': '1997-05-09T00:00:00.000Z',
    'ShippedDate': '1997-05-12T00:00:00.000Z',
    'Freight': 74.46,
    'ShipName': 'Eastern Connection',
    'ShipAddress': '35 King George',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10533,
    'CustomerID': 'FOLKO',
    'OrderDate': '1997-05-12T00:00:00.000Z',
    'ShippedDate': '1997-05-22T00:00:00.000Z',
    'Freight': 188.04,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10534,

```

```

      'CustomerID': 'LEHMS',
      'OrderDate': '1997-05-12T00:00:00.000Z',
      'ShippedDate': '1997-05-14T00:00:00.000Z',
      'Freight': 27.94,
      'ShipName': 'Lehmanns Marktstand',
      'ShipAddress': 'Magazinweg 7',
      'ShipCity': 'Frankfurt a.M.',
      'ShipRegion': null,
      'ShipCountry': 'Germany'
    },
    {
      'OrderID': 10535,
      'CustomerID': 'ANTON',
      'OrderDate': '1997-05-13T00:00:00.000Z',
      'ShippedDate': '1997-05-21T00:00:00.000Z',
      'Freight': 15.64,
      'ShipName': 'Antonio Moreno Taquería',
      'ShipAddress': 'Mataderos 2312',
      'ShipCity': 'México D.F.',
      'ShipRegion': null,
      'ShipCountry': 'Mexico'
    },
    {
      'OrderID': 10536,
      'CustomerID': 'LEHMS',
      'OrderDate': '1997-05-14T00:00:00.000Z',
      'ShippedDate': '1997-06-06T00:00:00.000Z',
      'Freight': 58.88,
      'ShipName': 'Lehmanns Marktstand',
      'ShipAddress': 'Magazinweg 7',
      'ShipCity': 'Frankfurt a.M.',
      'ShipRegion': null,
      'ShipCountry': 'Germany'
    },
    {
      'OrderID': 10537,
      'CustomerID': 'RICSU',
      'OrderDate': '1997-05-14T00:00:00.000Z',
      'ShippedDate': '1997-05-19T00:00:00.000Z',
      'Freight': 78.85,
      'ShipName': 'Richter Supermarkt',
      'ShipAddress': 'Starenweg 5',
      'ShipCity': 'Genève',
      'ShipRegion': null,
      'ShipCountry': 'Switzerland'
    },
    {
      'OrderID': 10538,
      'CustomerID': 'BSBEV',
      'OrderDate': '1997-05-15T00:00:00.000Z',
      'ShippedDate': '1997-05-16T00:00:00.000Z',
      'Freight': 4.87,
      'ShipName': 'B\' Beverages',
      'ShipAddress': 'Fauntleroy Circus',
      'ShipCity': 'London',
      'ShipRegion': null,
      'ShipCountry': 'UK'
    }
  ]

```

```

},
{
  'OrderID': 10539,
  'CustomerID': 'BSBEV',
  'OrderDate': '1997-05-16T00:00:00.000Z',
  'ShippedDate': '1997-05-23T00:00:00.000Z',
  'Freight': 12.36,
  'ShipName': 'B\ Beverages',
  'ShipAddress': 'Fauntleroy Circus',
  'ShipCity': 'London',
  'ShipRegion': null,
  'ShipCountry': 'UK'
},
{
  'OrderID': 10540,
  'CustomerID': 'QUICK',
  'OrderDate': '1997-05-19T00:00:00.000Z',
  'ShippedDate': '1997-06-13T00:00:00.000Z',
  'Freight': 1007.64,
  'ShipName': 'QUICK-Stop',
  'ShipAddress': 'Taucherstraße 10',
  'ShipCity': 'Cunewalde',
  'ShipRegion': null,
  'ShipCountry': 'Germany'
},
{
  'OrderID': 10541,
  'CustomerID': 'HANAR',
  'OrderDate': '1997-05-19T00:00:00.000Z',
  'ShippedDate': '1997-05-29T00:00:00.000Z',
  'Freight': 68.65,
  'ShipName': 'Hanari Carnes',
  'ShipAddress': 'Rua do Paço, 67',
  'ShipCity': 'Rio de Janeiro',
  'ShipRegion': 'RJ',
  'ShipCountry': 'Brazil'
},
{
  'OrderID': 10542,
  'CustomerID': 'KOENE',
  'OrderDate': '1997-05-20T00:00:00.000Z',
  'ShippedDate': '1997-05-26T00:00:00.000Z',
  'Freight': 10.95,
  'ShipName': 'Königlich Essen',
  'ShipAddress': 'Maubelstr. 90',
  'ShipCity': 'Brandenburg',
  'ShipRegion': null,
  'ShipCountry': 'Germany'
},
{
  'OrderID': 10543,
  'CustomerID': 'LILAS',
  'OrderDate': '1997-05-21T00:00:00.000Z',
  'ShippedDate': '1997-05-23T00:00:00.000Z',
  'Freight': 48.17,
  'ShipName': 'LILA-Supermercado',
  'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',

```

```

    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10544,
    'CustomerID': 'LONEP',
    'OrderDate': '1997-05-21T00:00:00.000Z',
    'ShippedDate': '1997-05-30T00:00:00.000Z',
    'Freight': 24.91,
    'ShipName': 'Lonesome Pine Restaurant',
    'ShipAddress': '89 Chiaroscuro Rd.',
    'ShipCity': 'Portland',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10545,
    'CustomerID': 'LAZYK',
    'OrderDate': '1997-05-22T00:00:00.000Z',
    'ShippedDate': '1997-06-26T00:00:00.000Z',
    'Freight': 11.92,
    'ShipName': 'Lazy K Kountry Store',
    'ShipAddress': '12 Orchestra Terrace',
    'ShipCity': 'Walla Walla',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10546,
    'CustomerID': 'VICTE',
    'OrderDate': '1997-05-23T00:00:00.000Z',
    'ShippedDate': '1997-05-27T00:00:00.000Z',
    'Freight': 194.72,
    'ShipName': 'Victuailles en stock',
    'ShipAddress': '2, rue du Commerce',
    'ShipCity': 'Lyon',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10547,
    'CustomerID': 'SEVES',
    'OrderDate': '1997-05-23T00:00:00.000Z',
    'ShippedDate': '1997-06-02T00:00:00.000Z',
    'Freight': 178.43,
    'ShipName': 'Seven Seas Imports',
    'ShipAddress': '90 Wadhurst Rd.',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10548,
    'CustomerID': 'TOMSP',
    'OrderDate': '1997-05-26T00:00:00.000Z',
    'ShippedDate': '1997-06-02T00:00:00.000Z',

```

```

    'Freight': 1.43,
    'ShipName': 'Toms Spezialitäten',
    'ShipAddress': 'Luisenstr. 48',
    'ShipCity': 'Münster',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10549,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-05-27T00:00:00.000Z',
    'ShippedDate': '1997-05-30T00:00:00.000Z',
    'Freight': 171.24,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10550,
    'CustomerID': 'GODOS',
    'OrderDate': '1997-05-28T00:00:00.000Z',
    'ShippedDate': '1997-06-06T00:00:00.000Z',
    'Freight': 4.32,
    'ShipName': 'Godos Cocina Típica',
    'ShipAddress': 'C/ Romero, 33',
    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10551,
    'CustomerID': 'FURIB',
    'OrderDate': '1997-05-28T00:00:00.000Z',
    'ShippedDate': '1997-06-06T00:00:00.000Z',
    'Freight': 72.95,
    'ShipName': 'Furia Bacalhau e Frutos do Mar',
    'ShipAddress': 'Jardim das rosas n. 32',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,
    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 10552,
    'CustomerID': 'HILAA',
    'OrderDate': '1997-05-29T00:00:00.000Z',
    'ShippedDate': '1997-06-05T00:00:00.000Z',
    'Freight': 83.22,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10553,

```

```

      'CustomerID': 'WARTH',
      'OrderDate': '1997-05-30T00:00:00.000Z',
      'ShippedDate': '1997-06-03T00:00:00.000Z',
      'Freight': 149.49,
      'ShipName': 'Wartian Herkku',
      'ShipAddress': 'Torikatu 38',
      'ShipCity': 'Oulu',
      'ShipRegion': null,
      'ShipCountry': 'Finland'
    },
    {
      'OrderID': 10554,
      'CustomerID': 'OTTIK',
      'OrderDate': '1997-05-30T00:00:00.000Z',
      'ShippedDate': '1997-06-05T00:00:00.000Z',
      'Freight': 120.97,
      'ShipName': 'Ottilies Käseladen',
      'ShipAddress': 'Mehrheimerstr. 369',
      'ShipCity': 'Köln',
      'ShipRegion': null,
      'ShipCountry': 'Germany'
    },
    {
      'OrderID': 10555,
      'CustomerID': 'SAVEA',
      'OrderDate': '1997-06-02T00:00:00.000Z',
      'ShippedDate': '1997-06-04T00:00:00.000Z',
      'Freight': 252.49,
      'ShipName': 'Save-a-lot Markets',
      'ShipAddress': '187 Suffolk Ln.',
      'ShipCity': 'Boise',
      'ShipRegion': 'ID',
      'ShipCountry': 'USA'
    },
    {
      'OrderID': 10556,
      'CustomerID': 'SIMOB',
      'OrderDate': '1997-06-03T00:00:00.000Z',
      'ShippedDate': '1997-06-13T00:00:00.000Z',
      'Freight': 9.8,
      'ShipName': 'Simons bistro',
      'ShipAddress': 'Vinbæltet 34',
      'ShipCity': 'Kobenhavn',
      'ShipRegion': null,
      'ShipCountry': 'Denmark'
    },
    {
      'OrderID': 10557,
      'CustomerID': 'LEHMS',
      'OrderDate': '1997-06-03T00:00:00.000Z',
      'ShippedDate': '1997-06-06T00:00:00.000Z',
      'Freight': 96.72,
      'ShipName': 'Lehmanns Marktstand',
      'ShipAddress': 'Magazinweg 7',
      'ShipCity': 'Frankfurt a.M.',
      'ShipRegion': null,
      'ShipCountry': 'Germany'
    }
  ]

```

```

},
{
  'OrderID': 10558,
  'CustomerID': 'AROUT',
  'OrderDate': '1997-06-04T00:00:00.000Z',
  'ShippedDate': '1997-06-10T00:00:00.000Z',
  'Freight': 72.97,
  'ShipName': 'Around the Horn',
  'ShipAddress': 'Brook Farm Stratford St. Mary',
  'ShipCity': 'Colchester',
  'ShipRegion': 'Essex',
  'ShipCountry': 'UK'
},
{
  'OrderID': 10559,
  'CustomerID': 'BLONP',
  'OrderDate': '1997-06-05T00:00:00.000Z',
  'ShippedDate': '1997-06-13T00:00:00.000Z',
  'Freight': 8.05,
  'ShipName': 'Blondel père et fils',
  'ShipAddress': '24, place Kléber',
  'ShipCity': 'Strasbourg',
  'ShipRegion': null,
  'ShipCountry': 'France'
},
{
  'OrderID': 10560,
  'CustomerID': 'FRANK',
  'OrderDate': '1997-06-06T00:00:00.000Z',
  'ShippedDate': '1997-06-09T00:00:00.000Z',
  'Freight': 36.65,
  'ShipName': 'Frankenversand',
  'ShipAddress': 'Berliner Platz 43',
  'ShipCity': 'München',
  'ShipRegion': null,
  'ShipCountry': 'Germany'
},
{
  'OrderID': 10561,
  'CustomerID': 'FOLKO',
  'OrderDate': '1997-06-06T00:00:00.000Z',
  'ShippedDate': '1997-06-09T00:00:00.000Z',
  'Freight': 242.21,
  'ShipName': 'Folk och fä HB',
  'ShipAddress': 'Åkergatan 24',
  'ShipCity': 'Bräcke',
  'ShipRegion': null,
  'ShipCountry': 'Sweden'
},
{
  'OrderID': 10562,
  'CustomerID': 'REGGC',
  'OrderDate': '1997-06-09T00:00:00.000Z',
  'ShippedDate': '1997-06-12T00:00:00.000Z',
  'Freight': 22.95,
  'ShipName': 'Reggiani Caseifici',
  'ShipAddress': 'Strada Provinciale 124',

```

```

    'ShipCity': 'Reggio Emilia',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10563,
    'CustomerID': 'RICAR',
    'OrderDate': '1997-06-10T00:00:00.000Z',
    'ShippedDate': '1997-06-24T00:00:00.000Z',
    'Freight': 60.43,
    'ShipName': 'Ricardo Adocicados',
    'ShipAddress': 'Av. Copacabana, 267',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10564,
    'CustomerID': 'RATTC',
    'OrderDate': '1997-06-10T00:00:00.000Z',
    'ShippedDate': '1997-06-16T00:00:00.000Z',
    'Freight': 13.75,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10565,
    'CustomerID': 'MEREPA',
    'OrderDate': '1997-06-11T00:00:00.000Z',
    'ShippedDate': '1997-06-18T00:00:00.000Z',
    'Freight': 7.15,
    'ShipName': 'Mère Paillarde',
    'ShipAddress': '43 rue St. Laurent',
    'ShipCity': 'Montréal',
    'ShipRegion': 'Québec',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10566,
    'CustomerID': 'BLONP',
    'OrderDate': '1997-06-12T00:00:00.000Z',
    'ShippedDate': '1997-06-18T00:00:00.000Z',
    'Freight': 88.4,
    'ShipName': 'Blondel père et fils',
    'ShipAddress': '24, place Kléber',
    'ShipCity': 'Strasbourg',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10567,
    'CustomerID': 'HUNGO',
    'OrderDate': '1997-06-12T00:00:00.000Z',
    'ShippedDate': '1997-06-17T00:00:00.000Z',

```



```

    'Freight': 33.97,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10568,
    'CustomerID': 'GALED',
    'OrderDate': '1997-06-13T00:00:00.000Z',
    'ShippedDate': '1997-07-09T00:00:00.000Z',
    'Freight': 6.54,
    'ShipName': 'Galería del gastronómo',
    'ShipAddress': 'Rambla de Cataluña, 23',
    'ShipCity': 'Barcelona',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10569,
    'CustomerID': 'RATTC',
    'OrderDate': '1997-06-16T00:00:00.000Z',
    'ShippedDate': '1997-07-11T00:00:00.000Z',
    'Freight': 58.98,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10570,
    'CustomerID': 'MEREP',
    'OrderDate': '1997-06-17T00:00:00.000Z',
    'ShippedDate': '1997-06-19T00:00:00.000Z',
    'Freight': 188.99,
    'ShipName': 'Mère Paillarde',
    'ShipAddress': '43 rue St. Laurent',
    'ShipCity': 'Montréal',
    'ShipRegion': 'Québec',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10571,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-06-17T00:00:00.000Z',
    'ShippedDate': '1997-07-04T00:00:00.000Z',
    'Freight': 26.06,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10572,

```

```

      'CustomerID': 'BERGS',
      'OrderDate': '1997-06-18T00:00:00.000Z',
      'ShippedDate': '1997-06-25T00:00:00.000Z',
      'Freight': 116.43,
      'ShipName': 'Berglunds snabbköp',
      'ShipAddress': 'Berguvsvägen 8',
      'ShipCity': 'Luleå',
      'ShipRegion': null,
      'ShipCountry': 'Sweden'
    },
    {
      'OrderID': 10573,
      'CustomerID': 'ANTON',
      'OrderDate': '1997-06-19T00:00:00.000Z',
      'ShippedDate': '1997-06-20T00:00:00.000Z',
      'Freight': 84.84,
      'ShipName': 'Antonio Moreno Taquería',
      'ShipAddress': 'Mataderos 2312',
      'ShipCity': 'México D.F.',
      'ShipRegion': null,
      'ShipCountry': 'Mexico'
    },
    {
      'OrderID': 10574,
      'CustomerID': 'TRAIH',
      'OrderDate': '1997-06-19T00:00:00.000Z',
      'ShippedDate': '1997-06-30T00:00:00.000Z',
      'Freight': 37.6,
      'ShipName': 'Trail\ Head Gourmet Provisioners',
      'ShipAddress': '722 DaVinci Blvd.',
      'ShipCity': 'Kirkland',
      'ShipRegion': 'WA',
      'ShipCountry': 'USA'
    },
    {
      'OrderID': 10575,
      'CustomerID': 'MORGK',
      'OrderDate': '1997-06-20T00:00:00.000Z',
      'ShippedDate': '1997-06-30T00:00:00.000Z',
      'Freight': 127.34,
      'ShipName': 'Morgenstern Gesundkost',
      'ShipAddress': 'Heerstr. 22',
      'ShipCity': 'Leipzig',
      'ShipRegion': null,
      'ShipCountry': 'Germany'
    },
    {
      'OrderID': 10576,
      'CustomerID': 'TORTU',
      'OrderDate': '1997-06-23T00:00:00.000Z',
      'ShippedDate': '1997-06-30T00:00:00.000Z',
      'Freight': 18.56,
      'ShipName': 'Tortuga Restaurante',
      'ShipAddress': 'Avda. Azteca 123',
      'ShipCity': 'México D.F.',
      'ShipRegion': null,
      'ShipCountry': 'Mexico'
    }
  ]

```

```

},
{
  'OrderID': 10577,
  'CustomerID': 'TRAIH',
  'OrderDate': '1997-06-23T00:00:00.000Z',
  'ShippedDate': '1997-06-30T00:00:00.000Z',
  'Freight': 25.41,
  'ShipName': 'Trail\' Head Gourmet Provisioners',
  'ShipAddress': '722 DaVinci Blvd.',
  'ShipCity': 'Kirkland',
  'ShipRegion': 'WA',
  'ShipCountry': 'USA'
},
{
  'OrderID': 10578,
  'CustomerID': 'BSBEV',
  'OrderDate': '1997-06-24T00:00:00.000Z',
  'ShippedDate': '1997-07-25T00:00:00.000Z',
  'Freight': 29.6,
  'ShipName': 'B\' Beverages',
  'ShipAddress': 'Fauntleroy Circus',
  'ShipCity': 'London',
  'ShipRegion': null,
  'ShipCountry': 'UK'
},
{
  'OrderID': 10579,
  'CustomerID': 'LETSS',
  'OrderDate': '1997-06-25T00:00:00.000Z',
  'ShippedDate': '1997-07-04T00:00:00.000Z',
  'Freight': 13.73,
  'ShipName': 'Let\' Stop N Shop',
  'ShipAddress': '87 Polk St. Suite 5',
  'ShipCity': 'San Francisco',
  'ShipRegion': 'CA',
  'ShipCountry': 'USA'
},
{
  'OrderID': 10580,
  'CustomerID': 'OTTIK',
  'OrderDate': '1997-06-26T00:00:00.000Z',
  'ShippedDate': '1997-07-01T00:00:00.000Z',
  'Freight': 75.89,
  'ShipName': 'Ottilies Käseladen',
  'ShipAddress': 'Mehrheimerstr. 369',
  'ShipCity': 'Köln',
  'ShipRegion': null,
  'ShipCountry': 'Germany'
},
{
  'OrderID': 10581,
  'CustomerID': 'FAMIA',
  'OrderDate': '1997-06-26T00:00:00.000Z',
  'ShippedDate': '1997-07-02T00:00:00.000Z',
  'Freight': 3.01,
  'ShipName': 'Familia Arquibaldo',
  'ShipAddress': 'Rua Orós, 92',

```

```

    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10582,
    'CustomerID': 'BLAUS',
    'OrderDate': '1997-06-27T00:00:00.000Z',
    'ShippedDate': '1997-07-14T00:00:00.000Z',
    'Freight': 27.71,
    'ShipName': 'Blauer See Delikatessen',
    'ShipAddress': 'Forsterstr. 57',
    'ShipCity': 'Mannheim',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10583,
    'CustomerID': 'WARTH',
    'OrderDate': '1997-06-30T00:00:00.000Z',
    'ShippedDate': '1997-07-04T00:00:00.000Z',
    'Freight': 7.28,
    'ShipName': 'Wartian Herkku',
    'ShipAddress': 'Torikatu 38',
    'ShipCity': 'Oulu',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10584,
    'CustomerID': 'BLONP',
    'OrderDate': '1997-06-30T00:00:00.000Z',
    'ShippedDate': '1997-07-04T00:00:00.000Z',
    'Freight': 59.14,
    'ShipName': 'Blondel père et fils',
    'ShipAddress': '24, place Kléber',
    'ShipCity': 'Strasbourg',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10585,
    'CustomerID': 'WELLI',
    'OrderDate': '1997-07-01T00:00:00.000Z',
    'ShippedDate': '1997-07-10T00:00:00.000Z',
    'Freight': 13.41,
    'ShipName': 'Wellington Importadora',
    'ShipAddress': 'Rua do Mercado, 12',
    'ShipCity': 'Resende',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10586,
    'CustomerID': 'REGGC',
    'OrderDate': '1997-07-02T00:00:00.000Z',
    'ShippedDate': '1997-07-09T00:00:00.000Z',

```

```

    'Freight': 0.48,
    'ShipName': 'Reggiani Caseifici',
    'ShipAddress': 'Strada Provinciale 124',
    'ShipCity': 'Reggio Emilia',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10587,
    'CustomerID': 'QUEDE',
    'OrderDate': '1997-07-02T00:00:00.000Z',
    'ShippedDate': '1997-07-09T00:00:00.000Z',
    'Freight': 62.52,
    'ShipName': 'Que Delícia',
    'ShipAddress': 'Rua da Panificadora, 12',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10588,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-07-03T00:00:00.000Z',
    'ShippedDate': '1997-07-10T00:00:00.000Z',
    'Freight': 194.67,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10589,
    'CustomerID': 'GREAL',
    'OrderDate': '1997-07-04T00:00:00.000Z',
    'ShippedDate': '1997-07-14T00:00:00.000Z',
    'Freight': 4.42,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10590,
    'CustomerID': 'MEREP',
    'OrderDate': '1997-07-07T00:00:00.000Z',
    'ShippedDate': '1997-07-14T00:00:00.000Z',
    'Freight': 44.77,
    'ShipName': 'Mère Paillarde',
    'ShipAddress': '43 rue St. Laurent',
    'ShipCity': 'Montréal',
    'ShipRegion': 'Québec',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10591,

```

```

    'CustomerID': 'VAFPE',
    'OrderDate': '1997-07-07T00:00:00.000Z',
    'ShippedDate': '1997-07-16T00:00:00.000Z',
    'Freight': 55.92,
    'ShipName': 'Vaffeljernet',
    'ShipAddress': 'Smagsloget 45',
    'ShipCity': 'Århus',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10592,
    'CustomerID': 'LEHMS',
    'OrderDate': '1997-07-08T00:00:00.000Z',
    'ShippedDate': '1997-07-16T00:00:00.000Z',
    'Freight': 32.1,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10593,
    'CustomerID': 'LEHMS',
    'OrderDate': '1997-07-09T00:00:00.000Z',
    'ShippedDate': '1997-08-13T00:00:00.000Z',
    'Freight': 174.2,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10594,
    'CustomerID': 'OLDWO',
    'OrderDate': '1997-07-09T00:00:00.000Z',
    'ShippedDate': '1997-07-16T00:00:00.000Z',
    'Freight': 5.24,
    'ShipName': 'Old World Delicatessen',
    'ShipAddress': '2743 Bering St.',
    'ShipCity': 'Anchorage',
    'ShipRegion': 'AK',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10595,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-07-10T00:00:00.000Z',
    'ShippedDate': '1997-07-14T00:00:00.000Z',
    'Freight': 96.78,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  }

```

```

    },
    {
        'OrderID': 10596,
        'CustomerID': 'WHITC',
        'OrderDate': '1997-07-11T00:00:00.000Z',
        'ShippedDate': '1997-08-12T00:00:00.000Z',
        'Freight': 16.34,
        'ShipName': 'White Clover Markets',
        'ShipAddress': '1029 - 12th Ave. S.',
        'ShipCity': 'Seattle',
        'ShipRegion': 'WA',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10597,
        'CustomerID': 'PICCO',
        'OrderDate': '1997-07-11T00:00:00.000Z',
        'ShippedDate': '1997-07-18T00:00:00.000Z',
        'Freight': 35.12,
        'ShipName': 'Piccolo und mehr',
        'ShipAddress': 'Geislweg 14',
        'ShipCity': 'Salzburg',
        'ShipRegion': null,
        'ShipCountry': 'Austria'
    },
    {
        'OrderID': 10598,
        'CustomerID': 'RATTC',
        'OrderDate': '1997-07-14T00:00:00.000Z',
        'ShippedDate': '1997-07-18T00:00:00.000Z',
        'Freight': 44.42,
        'ShipName': 'Rattlesnake Canyon Grocery',
        'ShipAddress': '2817 Milton Dr.',
        'ShipCity': 'Albuquerque',
        'ShipRegion': 'NM',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10599,
        'CustomerID': 'BSBEV',
        'OrderDate': '1997-07-15T00:00:00.000Z',
        'ShippedDate': '1997-07-21T00:00:00.000Z',
        'Freight': 29.98,
        'ShipName': 'B Beverages',
        'ShipAddress': 'Fauntleroy Circus',
        'ShipCity': 'London',
        'ShipRegion': null,
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10600,
        'CustomerID': 'HUNGC',
        'OrderDate': '1997-07-16T00:00:00.000Z',
        'ShippedDate': '1997-07-21T00:00:00.000Z',
        'Freight': 45.13,
        'ShipName': 'Hungry Coyote Import Store',
        'ShipAddress': 'City Center Plaza 516 Main St.',
    }

```

```

        'ShipCity': 'Elgin',
        'ShipRegion': 'OR',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10601,
        'CustomerID': 'HILAA',
        'OrderDate': '1997-07-16T00:00:00.000Z',
        'ShippedDate': '1997-07-22T00:00:00.000Z',
        'Freight': 58.3,
        'ShipName': 'HILARION-Abastos',
        'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
        'ShipCity': 'San Cristóbal',
        'ShipRegion': 'Táchira',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10602,
        'CustomerID': 'VAFFE',
        'OrderDate': '1997-07-17T00:00:00.000Z',
        'ShippedDate': '1997-07-22T00:00:00.000Z',
        'Freight': 2.92,
        'ShipName': 'Vaffeljernet',
        'ShipAddress': 'Smagsloget 45',
        'ShipCity': 'Århus',
        'ShipRegion': null,
        'ShipCountry': 'Denmark'
    },
    {
        'OrderID': 10603,
        'CustomerID': 'SAVEA',
        'OrderDate': '1997-07-18T00:00:00.000Z',
        'ShippedDate': '1997-08-08T00:00:00.000Z',
        'Freight': 48.77,
        'ShipName': 'Save-a-lot Markets',
        'ShipAddress': '187 Suffolk Ln.',
        'ShipCity': 'Boise',
        'ShipRegion': 'ID',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10604,
        'CustomerID': 'FURIB',
        'OrderDate': '1997-07-18T00:00:00.000Z',
        'ShippedDate': '1997-07-29T00:00:00.000Z',
        'Freight': 7.46,
        'ShipName': 'Furia Bacalhau e Frutos do Mar',
        'ShipAddress': 'Jardim das rosas n. 32',
        'ShipCity': 'Lisboa',
        'ShipRegion': null,
        'ShipCountry': 'Portugal'
    },
    {
        'OrderID': 10605,
        'CustomerID': 'MEREP',
        'OrderDate': '1997-07-21T00:00:00.000Z',
        'ShippedDate': '1997-07-29T00:00:00.000Z',

```



```

    'Freight': 379.13,
    'ShipName': 'Mère Paillarde',
    'ShipAddress': '43 rue St. Laurent',
    'ShipCity': 'Montréal',
    'ShipRegion': 'Québec',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10606,
    'CustomerID': 'TRADH',
    'OrderDate': '1997-07-22T00:00:00.000Z',
    'ShippedDate': '1997-07-31T00:00:00.000Z',
    'Freight': 79.4,
    'ShipName': 'Tradiçao Hipermercados',
    'ShipAddress': 'Av. Inês de Castro, 414',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10607,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-07-22T00:00:00.000Z',
    'ShippedDate': '1997-07-25T00:00:00.000Z',
    'Freight': 200.24,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10608,
    'CustomerID': 'TOMSP',
    'OrderDate': '1997-07-23T00:00:00.000Z',
    'ShippedDate': '1997-08-01T00:00:00.000Z',
    'Freight': 27.79,
    'ShipName': 'Toms Spezialitäten',
    'ShipAddress': 'Luisenstr. 48',
    'ShipCity': 'Münster',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10609,
    'CustomerID': 'DUMON',
    'OrderDate': '1997-07-24T00:00:00.000Z',
    'ShippedDate': '1997-07-30T00:00:00.000Z',
    'Freight': 1.85,
    'ShipName': 'Du monde entier',
    'ShipAddress': '67, rue des Cinquante Otages',
    'ShipCity': 'Nantes',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10610,

```

```

    'CustomerID': 'LAMAI',
    'OrderDate': '1997-07-25T00:00:00.000Z',
    'ShippedDate': '1997-08-06T00:00:00.000Z',
    'Freight': 26.78,
    'ShipName': 'La maison d\'Asie',
    'ShipAddress': '1 rue Alsace-Lorraine',
    'ShipCity': 'Toulouse',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10611,
    'CustomerID': 'WOLZA',
    'OrderDate': '1997-07-25T00:00:00.000Z',
    'ShippedDate': '1997-08-01T00:00:00.000Z',
    'Freight': 80.65,
    'ShipName': 'Wolski Zajazd',
    'ShipAddress': 'ul. Filtrowa 68',
    'ShipCity': 'Warszawa',
    'ShipRegion': null,
    'ShipCountry': 'Poland'
  },
  {
    'OrderID': 10612,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-07-28T00:00:00.000Z',
    'ShippedDate': '1997-08-01T00:00:00.000Z',
    'Freight': 544.08,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10613,
    'CustomerID': 'HILAA',
    'OrderDate': '1997-07-29T00:00:00.000Z',
    'ShippedDate': '1997-08-01T00:00:00.000Z',
    'Freight': 8.11,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10614,
    'CustomerID': 'BLAUS',
    'OrderDate': '1997-07-29T00:00:00.000Z',
    'ShippedDate': '1997-08-01T00:00:00.000Z',
    'Freight': 1.93,
    'ShipName': 'Blauer See Delikatessen',
    'ShipAddress': 'Forsterstr. 57',
    'ShipCity': 'Mannheim',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  }

```

```

    },
    {
        'OrderID': 10615,
        'CustomerID': 'WILMK',
        'OrderDate': '1997-07-30T00:00:00.000Z',
        'ShippedDate': '1997-08-06T00:00:00.000Z',
        'Freight': 0.75,
        'ShipName': 'Wilman Kala',
        'ShipAddress': 'Keskuskatu 45',
        'ShipCity': 'Helsinki',
        'ShipRegion': null,
        'ShipCountry': 'Finland'
    },
    {
        'OrderID': 10616,
        'CustomerID': 'GREAL',
        'OrderDate': '1997-07-31T00:00:00.000Z',
        'ShippedDate': '1997-08-05T00:00:00.000Z',
        'Freight': 116.53,
        'ShipName': 'Great Lakes Food Market',
        'ShipAddress': '2732 Baker Blvd.',
        'ShipCity': 'Eugene',
        'ShipRegion': 'OR',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10617,
        'CustomerID': 'GREAL',
        'OrderDate': '1997-07-31T00:00:00.000Z',
        'ShippedDate': '1997-08-04T00:00:00.000Z',
        'Freight': 18.53,
        'ShipName': 'Great Lakes Food Market',
        'ShipAddress': '2732 Baker Blvd.',
        'ShipCity': 'Eugene',
        'ShipRegion': 'OR',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10618,
        'CustomerID': 'MEREP',
        'OrderDate': '1997-08-01T00:00:00.000Z',
        'ShippedDate': '1997-08-08T00:00:00.000Z',
        'Freight': 154.68,
        'ShipName': 'Mère Paillarde',
        'ShipAddress': '43 rue St. Laurent',
        'ShipCity': 'Montréal',
        'ShipRegion': 'Québec',
        'ShipCountry': 'Canada'
    },
    {
        'OrderID': 10619,
        'CustomerID': 'MEREP',
        'OrderDate': '1997-08-04T00:00:00.000Z',
        'ShippedDate': '1997-08-07T00:00:00.000Z',
        'Freight': 91.05,
        'ShipName': 'Mère Paillarde',
        'ShipAddress': '43 rue St. Laurent',
    }

```

```

        'ShipCity': 'Montréal',
        'ShipRegion': 'Québec',
        'ShipCountry': 'Canada'
    },
    {
        'OrderID': 10620,
        'CustomerID': 'LAUGB',
        'OrderDate': '1997-08-05T00:00:00.000Z',
        'ShippedDate': '1997-08-14T00:00:00.000Z',
        'Freight': 0.94,
        'ShipName': 'Laughing Bacchus Wine Cellars',
        'ShipAddress': '2319 Elm St.',
        'ShipCity': 'Vancouver',
        'ShipRegion': 'BC',
        'ShipCountry': 'Canada'
    },
    {
        'OrderID': 10621,
        'CustomerID': 'ISLAT',
        'OrderDate': '1997-08-05T00:00:00.000Z',
        'ShippedDate': '1997-08-11T00:00:00.000Z',
        'Freight': 23.73,
        'ShipName': 'Island Trading',
        'ShipAddress': 'Garden House Crowther Way',
        'ShipCity': 'Cowes',
        'ShipRegion': 'Isle of Wight',
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10622,
        'CustomerID': 'RICAR',
        'OrderDate': '1997-08-06T00:00:00.000Z',
        'ShippedDate': '1997-08-11T00:00:00.000Z',
        'Freight': 50.97,
        'ShipName': 'Ricardo Adocicados',
        'ShipAddress': 'Av. Copacabana, 267',
        'ShipCity': 'Rio de Janeiro',
        'ShipRegion': 'RJ',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10623,
        'CustomerID': 'FRANK',
        'OrderDate': '1997-08-07T00:00:00.000Z',
        'ShippedDate': '1997-08-12T00:00:00.000Z',
        'Freight': 97.18,
        'ShipName': 'Frankenversand',
        'ShipAddress': 'Berliner Platz 43',
        'ShipCity': 'München',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10624,
        'CustomerID': 'THECR',
        'OrderDate': '1997-08-07T00:00:00.000Z',
        'ShippedDate': '1997-08-19T00:00:00.000Z',

```

```

    'Freight': 94.8,
    'ShipName': 'The Cracker Box',
    'ShipAddress': '55 Grizzly Peak Rd.',
    'ShipCity': 'Butte',
    'ShipRegion': 'MT',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10625,
    'CustomerID': 'ANATR',
    'OrderDate': '1997-08-08T00:00:00.000Z',
    'ShippedDate': '1997-08-14T00:00:00.000Z',
    'Freight': 43.9,
    'ShipName': 'Ana Trujillo Emparedados y helados',
    'ShipAddress': 'Avda. de la Constitución 2222',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10626,
    'CustomerID': 'BERGS',
    'OrderDate': '1997-08-11T00:00:00.000Z',
    'ShippedDate': '1997-08-20T00:00:00.000Z',
    'Freight': 138.69,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10627,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-08-11T00:00:00.000Z',
    'ShippedDate': '1997-08-21T00:00:00.000Z',
    'Freight': 107.46,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10628,
    'CustomerID': 'BLONP',
    'OrderDate': '1997-08-12T00:00:00.000Z',
    'ShippedDate': '1997-08-20T00:00:00.000Z',
    'Freight': 30.36,
    'ShipName': 'Blondel père et fils',
    'ShipAddress': '24, place Kléber',
    'ShipCity': 'Strasbourg',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10629,

```

```

    'CustomerID': 'GODOS',
    'OrderDate': '1997-08-12T00:00:00.000Z',
    'ShippedDate': '1997-08-20T00:00:00.000Z',
    'Freight': 85.46,
    'ShipName': 'Godos Cocina Típica',
    'ShipAddress': 'C/ Romero, 33',
    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10630,
    'CustomerID': 'KOENE',
    'OrderDate': '1997-08-13T00:00:00.000Z',
    'ShippedDate': '1997-08-19T00:00:00.000Z',
    'Freight': 32.35,
    'ShipName': 'Königlich Essen',
    'ShipAddress': 'Maubelstr. 90',
    'ShipCity': 'Brandenburg',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10631,
    'CustomerID': 'LAMAI',
    'OrderDate': '1997-08-14T00:00:00.000Z',
    'ShippedDate': '1997-08-15T00:00:00.000Z',
    'Freight': 0.87,
    'ShipName': 'La maison d\'Asie',
    'ShipAddress': '1 rue Alsace-Lorraine',
    'ShipCity': 'Toulouse',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10632,
    'CustomerID': 'WANDK',
    'OrderDate': '1997-08-14T00:00:00.000Z',
    'ShippedDate': '1997-08-19T00:00:00.000Z',
    'Freight': 41.38,
    'ShipName': 'Die Wandernde Kuh',
    'ShipAddress': 'Adenauerallee 900',
    'ShipCity': 'Stuttgart',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10633,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-08-15T00:00:00.000Z',
    'ShippedDate': '1997-08-18T00:00:00.000Z',
    'Freight': 477.9,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  }

```

```

    },
    {
        'OrderID': 10634,
        'CustomerID': 'FOLIG',
        'OrderDate': '1997-08-15T00:00:00.000Z',
        'ShippedDate': '1997-08-21T00:00:00.000Z',
        'Freight': 487.38,
        'ShipName': 'Folies gourmandes',
        'ShipAddress': '184, chaussée de Tournai',
        'ShipCity': 'Lille',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10635,
        'CustomerID': 'MAGAA',
        'OrderDate': '1997-08-18T00:00:00.000Z',
        'ShippedDate': '1997-08-21T00:00:00.000Z',
        'Freight': 47.46,
        'ShipName': 'Magazzini Alimentari Riuniti',
        'ShipAddress': 'Via Ludovico il Moro 22',
        'ShipCity': 'Bergamo',
        'ShipRegion': null,
        'ShipCountry': 'Italy'
    },
    {
        'OrderID': 10636,
        'CustomerID': 'WARTH',
        'OrderDate': '1997-08-19T00:00:00.000Z',
        'ShippedDate': '1997-08-26T00:00:00.000Z',
        'Freight': 1.15,
        'ShipName': 'Wartian Herkku',
        'ShipAddress': 'Torikatu 38',
        'ShipCity': 'Oulu',
        'ShipRegion': null,
        'ShipCountry': 'Finland'
    },
    {
        'OrderID': 10637,
        'CustomerID': 'QUEEN',
        'OrderDate': '1997-08-19T00:00:00.000Z',
        'ShippedDate': '1997-08-26T00:00:00.000Z',
        'Freight': 201.29,
        'ShipName': 'Queen Cozinha',
        'ShipAddress': 'Alameda dos Canários, 891',
        'ShipCity': 'Sao Paulo',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10638,
        'CustomerID': 'LINOD',
        'OrderDate': '1997-08-20T00:00:00.000Z',
        'ShippedDate': '1997-09-01T00:00:00.000Z',
        'Freight': 158.44,
        'ShipName': 'LINO-Delicateses',
        'ShipAddress': 'Ave. 5 de Mayo Porlamar',

```

```

    'ShipCity': 'I. de Margarita',
    'ShipRegion': 'Nueva Esparta',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10639,
    'CustomerID': 'SANTG',
    'OrderDate': '1997-08-20T00:00:00.000Z',
    'ShippedDate': '1997-08-27T00:00:00.000Z',
    'Freight': 38.64,
    'ShipName': 'Santé Gourmet',
    'ShipAddress': 'Erling Skakkes gate 78',
    'ShipCity': 'Stavern',
    'ShipRegion': null,
    'ShipCountry': 'Norway'
  },
  {
    'OrderID': 10640,
    'CustomerID': 'WANDK',
    'OrderDate': '1997-08-21T00:00:00.000Z',
    'ShippedDate': '1997-08-28T00:00:00.000Z',
    'Freight': 23.55,
    'ShipName': 'Die Wandernde Kuh',
    'ShipAddress': 'Adenauerallee 900',
    'ShipCity': 'Stuttgart',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10641,
    'CustomerID': 'HILAA',
    'OrderDate': '1997-08-22T00:00:00.000Z',
    'ShippedDate': '1997-08-26T00:00:00.000Z',
    'Freight': 179.61,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10642,
    'CustomerID': 'SIMOB',
    'OrderDate': '1997-08-22T00:00:00.000Z',
    'ShippedDate': '1997-09-05T00:00:00.000Z',
    'Freight': 41.89,
    'ShipName': 'Simons bistro',
    'ShipAddress': 'Vinbæltet 34',
    'ShipCity': 'Kobenhavn',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10643,
    'CustomerID': 'ALFKI',
    'OrderDate': '1997-08-25T00:00:00.000Z',
    'ShippedDate': '1997-09-02T00:00:00.000Z',

```



```

    'Freight': 29.46,
    'ShipName': 'Alfreds Futterkiste',
    'ShipAddress': 'Obere Str. 57',
    'ShipCity': 'Berlin',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10644,
    'CustomerID': 'WELLI',
    'OrderDate': '1997-08-25T00:00:00.000Z',
    'ShippedDate': '1997-09-01T00:00:00.000Z',
    'Freight': 0.14,
    'ShipName': 'Wellington Importadora',
    'ShipAddress': 'Rua do Mercado, 12',
    'ShipCity': 'Resende',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10645,
    'CustomerID': 'HANAR',
    'OrderDate': '1997-08-26T00:00:00.000Z',
    'ShippedDate': '1997-09-02T00:00:00.000Z',
    'Freight': 12.41,
    'ShipName': 'Hanari Carnes',
    'ShipAddress': 'Rua do Paço, 67',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10646,
    'CustomerID': 'HUNGO',
    'OrderDate': '1997-08-27T00:00:00.000Z',
    'ShippedDate': '1997-09-03T00:00:00.000Z',
    'Freight': 142.33,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10647,
    'CustomerID': 'QUEDE',
    'OrderDate': '1997-08-27T00:00:00.000Z',
    'ShippedDate': '1997-09-03T00:00:00.000Z',
    'Freight': 45.54,
    'ShipName': 'Que Delícia',
    'ShipAddress': 'Rua da Panificadora, 12',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10648,

```

```

      'CustomerID': 'RICAR',
      'OrderDate': '1997-08-28T00:00:00.000Z',
      'ShippedDate': '1997-09-09T00:00:00.000Z',
      'Freight': 14.25,
      'ShipName': 'Ricardo Adocicados',
      'ShipAddress': 'Av. Copacabana, 267',
      'ShipCity': 'Rio de Janeiro',
      'ShipRegion': 'RJ',
      'ShipCountry': 'Brazil'
    },
    {
      'OrderID': 10649,
      'CustomerID': 'MAISD',
      'OrderDate': '1997-08-28T00:00:00.000Z',
      'ShippedDate': '1997-08-29T00:00:00.000Z',
      'Freight': 6.2,
      'ShipName': 'Maison Dewey',
      'ShipAddress': 'Rue Joseph-Bens 532',
      'ShipCity': 'Bruxelles',
      'ShipRegion': null,
      'ShipCountry': 'Belgium'
    },
    {
      'OrderID': 10650,
      'CustomerID': 'FAMIA',
      'OrderDate': '1997-08-29T00:00:00.000Z',
      'ShippedDate': '1997-09-03T00:00:00.000Z',
      'Freight': 176.81,
      'ShipName': 'Familia Arquibaldo',
      'ShipAddress': 'Rua Orós, 92',
      'ShipCity': 'Sao Paulo',
      'ShipRegion': 'SP',
      'ShipCountry': 'Brazil'
    },
    {
      'OrderID': 10651,
      'CustomerID': 'WANDK',
      'OrderDate': '1997-09-01T00:00:00.000Z',
      'ShippedDate': '1997-09-11T00:00:00.000Z',
      'Freight': 20.6,
      'ShipName': 'Die Wandernde Kuh',
      'ShipAddress': 'Adenauerallee 900',
      'ShipCity': 'Stuttgart',
      'ShipRegion': null,
      'ShipCountry': 'Germany'
    },
    {
      'OrderID': 10652,
      'CustomerID': 'GOURL',
      'OrderDate': '1997-09-01T00:00:00.000Z',
      'ShippedDate': '1997-09-08T00:00:00.000Z',
      'Freight': 7.14,
      'ShipName': 'Gourmet Lanchonetes',
      'ShipAddress': 'Av. Brasil, 442',
      'ShipCity': 'Campinas',
      'ShipRegion': 'SP',
      'ShipCountry': 'Brazil'
    }
  ]

```

```

    },
    {
        'OrderID': 10653,
        'CustomerID': 'FRANK',
        'OrderDate': '1997-09-02T00:00:00.000Z',
        'ShippedDate': '1997-09-19T00:00:00.000Z',
        'Freight': 93.25,
        'ShipName': 'Frankenversand',
        'ShipAddress': 'Berliner Platz 43',
        'ShipCity': 'München',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10654,
        'CustomerID': 'BERGS',
        'OrderDate': '1997-09-02T00:00:00.000Z',
        'ShippedDate': '1997-09-11T00:00:00.000Z',
        'Freight': 55.26,
        'ShipName': 'Berglunds snabbköp',
        'ShipAddress': 'Berguvsvägen 8',
        'ShipCity': 'Luleå',
        'ShipRegion': null,
        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 10655,
        'CustomerID': 'REGGC',
        'OrderDate': '1997-09-03T00:00:00.000Z',
        'ShippedDate': '1997-09-11T00:00:00.000Z',
        'Freight': 4.41,
        'ShipName': 'Reggiani Caseifici',
        'ShipAddress': 'Strada Provinciale 124',
        'ShipCity': 'Reggio Emilia',
        'ShipRegion': null,
        'ShipCountry': 'Italy'
    },
    {
        'OrderID': 10656,
        'CustomerID': 'GREAL',
        'OrderDate': '1997-09-04T00:00:00.000Z',
        'ShippedDate': '1997-09-10T00:00:00.000Z',
        'Freight': 57.15,
        'ShipName': 'Great Lakes Food Market',
        'ShipAddress': '2732 Baker Blvd.',
        'ShipCity': 'Eugene',
        'ShipRegion': 'OR',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10657,
        'CustomerID': 'SAVEA',
        'OrderDate': '1997-09-04T00:00:00.000Z',
        'ShippedDate': '1997-09-15T00:00:00.000Z',
        'Freight': 352.69,
        'ShipName': 'Save-a-lot Markets',
        'ShipAddress': '187 Suffolk Ln.',
    }

```

```

      'ShipCity': 'Boise',
      'ShipRegion': 'ID',
      'ShipCountry': 'USA'
    },
    {
      'OrderID': 10658,
      'CustomerID': 'QUICK',
      'OrderDate': '1997-09-05T00:00:00.000Z',
      'ShippedDate': '1997-09-08T00:00:00.000Z',
      'Freight': 364.15,
      'ShipName': 'QUICK-Stop',
      'ShipAddress': 'Taucherstraße 10',
      'ShipCity': 'Cunewalde',
      'ShipRegion': null,
      'ShipCountry': 'Germany'
    },
    {
      'OrderID': 10659,
      'CustomerID': 'QUEEN',
      'OrderDate': '1997-09-05T00:00:00.000Z',
      'ShippedDate': '1997-09-10T00:00:00.000Z',
      'Freight': 105.81,
      'ShipName': 'Queen Cozinha',
      'ShipAddress': 'Alameda dos Canários, 891',
      'ShipCity': 'Sao Paulo',
      'ShipRegion': 'SP',
      'ShipCountry': 'Brazil'
    },
    {
      'OrderID': 10660,
      'CustomerID': 'HUNGC',
      'OrderDate': '1997-09-08T00:00:00.000Z',
      'ShippedDate': '1997-10-15T00:00:00.000Z',
      'Freight': 111.29,
      'ShipName': 'Hungry Coyote Import Store',
      'ShipAddress': 'City Center Plaza 516 Main St.',
      'ShipCity': 'Elgin',
      'ShipRegion': 'OR',
      'ShipCountry': 'USA'
    },
    {
      'OrderID': 10661,
      'CustomerID': 'HUNGO',
      'OrderDate': '1997-09-09T00:00:00.000Z',
      'ShippedDate': '1997-09-15T00:00:00.000Z',
      'Freight': 17.55,
      'ShipName': 'Hungry Owl All-Night Grocers',
      'ShipAddress': '8 Johnstown Road',
      'ShipCity': 'Cork',
      'ShipRegion': 'Co. Cork',
      'ShipCountry': 'Ireland'
    },
    {
      'OrderID': 10662,
      'CustomerID': 'LONEP',
      'OrderDate': '1997-09-09T00:00:00.000Z',
      'ShippedDate': '1997-09-18T00:00:00.000Z',

```

```

    'Freight': 1.28,
    'ShipName': 'Lonesome Pine Restaurant',
    'ShipAddress': '89 Chiaroscuro Rd.',
    'ShipCity': 'Portland',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10663,
    'CustomerID': 'BONAP',
    'OrderDate': '1997-09-10T00:00:00.000Z',
    'ShippedDate': '1997-10-03T00:00:00.000Z',
    'Freight': 113.15,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10664,
    'CustomerID': 'FURIB',
    'OrderDate': '1997-09-10T00:00:00.000Z',
    'ShippedDate': '1997-09-19T00:00:00.000Z',
    'Freight': 1.27,
    'ShipName': 'Furia Bacalhau e Frutos do Mar',
    'ShipAddress': 'Jardim das rosas n. 32',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,
    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 10665,
    'CustomerID': 'LONEP',
    'OrderDate': '1997-09-11T00:00:00.000Z',
    'ShippedDate': '1997-09-17T00:00:00.000Z',
    'Freight': 26.31,
    'ShipName': 'Lonesome Pine Restaurant',
    'ShipAddress': '89 Chiaroscuro Rd.',
    'ShipCity': 'Portland',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10666,
    'CustomerID': 'RICSU',
    'OrderDate': '1997-09-12T00:00:00.000Z',
    'ShippedDate': '1997-09-22T00:00:00.000Z',
    'Freight': 232.42,
    'ShipName': 'Richter Supermarkt',
    'ShipAddress': 'Starenweg 5',
    'ShipCity': 'Genève',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10667,

```

```

    'CustomerID': 'ERNSH',
    'OrderDate': '1997-09-12T00:00:00.000Z',
    'ShippedDate': '1997-09-19T00:00:00.000Z',
    'Freight': 78.09,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10668,
    'CustomerID': 'WANDK',
    'OrderDate': '1997-09-15T00:00:00.000Z',
    'ShippedDate': '1997-09-23T00:00:00.000Z',
    'Freight': 47.22,
    'ShipName': 'Die Wandernde Kuh',
    'ShipAddress': 'Adenauerallee 900',
    'ShipCity': 'Stuttgart',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10669,
    'CustomerID': 'SIMOB',
    'OrderDate': '1997-09-15T00:00:00.000Z',
    'ShippedDate': '1997-09-22T00:00:00.000Z',
    'Freight': 24.39,
    'ShipName': 'Simons bistro',
    'ShipAddress': 'Vinbæltet 34',
    'ShipCity': 'Kobenhavn',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10670,
    'CustomerID': 'FRANK',
    'OrderDate': '1997-09-16T00:00:00.000Z',
    'ShippedDate': '1997-09-18T00:00:00.000Z',
    'Freight': 203.48,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10671,
    'CustomerID': 'FRANR',
    'OrderDate': '1997-09-17T00:00:00.000Z',
    'ShippedDate': '1997-09-24T00:00:00.000Z',
    'Freight': 30.34,
    'ShipName': 'France restauration',
    'ShipAddress': '54, rue Royale',
    'ShipCity': 'Nantes',
    'ShipRegion': null,
    'ShipCountry': 'France'
  }

```

```

    },
    {
        'OrderID': 10672,
        'CustomerID': 'BERGS',
        'OrderDate': '1997-09-17T00:00:00.000Z',
        'ShippedDate': '1997-09-26T00:00:00.000Z',
        'Freight': 95.75,
        'ShipName': 'Berglunds snabbköp',
        'ShipAddress': 'Berguvsvägen 8',
        'ShipCity': 'Luleå',
        'ShipRegion': null,
        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 10673,
        'CustomerID': 'WILMK',
        'OrderDate': '1997-09-18T00:00:00.000Z',
        'ShippedDate': '1997-09-19T00:00:00.000Z',
        'Freight': 22.76,
        'ShipName': 'Wilman Kala',
        'ShipAddress': 'Keskuskatu 45',
        'ShipCity': 'Helsinki',
        'ShipRegion': null,
        'ShipCountry': 'Finland'
    },
    {
        'OrderID': 10674,
        'CustomerID': 'ISLAT',
        'OrderDate': '1997-09-18T00:00:00.000Z',
        'ShippedDate': '1997-09-30T00:00:00.000Z',
        'Freight': 0.9,
        'ShipName': 'Island Trading',
        'ShipAddress': 'Garden House Crowther Way',
        'ShipCity': 'Cowes',
        'ShipRegion': 'Isle of Wight',
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10675,
        'CustomerID': 'FRANK',
        'OrderDate': '1997-09-19T00:00:00.000Z',
        'ShippedDate': '1997-09-23T00:00:00.000Z',
        'Freight': 31.85,
        'ShipName': 'Frankenversand',
        'ShipAddress': 'Berliner Platz 43',
        'ShipCity': 'München',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10676,
        'CustomerID': 'TORTU',
        'OrderDate': '1997-09-22T00:00:00.000Z',
        'ShippedDate': '1997-09-29T00:00:00.000Z',
        'Freight': 2.01,
        'ShipName': 'Tortuga Restaurante',
        'ShipAddress': 'Avda. Azteca 123',
    }

```

```

    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10677,
    'CustomerID': 'ANTON',
    'OrderDate': '1997-09-22T00:00:00.000Z',
    'ShippedDate': '1997-09-26T00:00:00.000Z',
    'Freight': 4.03,
    'ShipName': 'Antonio Moreno Taquería',
    'ShipAddress': 'Mataderos 2312',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10678,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-09-23T00:00:00.000Z',
    'ShippedDate': '1997-10-16T00:00:00.000Z',
    'Freight': 388.98,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10679,
    'CustomerID': 'BLONP',
    'OrderDate': '1997-09-23T00:00:00.000Z',
    'ShippedDate': '1997-09-30T00:00:00.000Z',
    'Freight': 27.94,
    'ShipName': 'Blondel père et fils',
    'ShipAddress': '24, place Kléber',
    'ShipCity': 'Strasbourg',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10680,
    'CustomerID': 'OLDWO',
    'OrderDate': '1997-09-24T00:00:00.000Z',
    'ShippedDate': '1997-09-26T00:00:00.000Z',
    'Freight': 26.61,
    'ShipName': 'Old World Delicatessen',
    'ShipAddress': '2743 Bering St.',
    'ShipCity': 'Anchorage',
    'ShipRegion': 'AK',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10681,
    'CustomerID': 'GREAL',
    'OrderDate': '1997-09-25T00:00:00.000Z',
    'ShippedDate': '1997-09-30T00:00:00.000Z',

```



```

    'Freight': 76.13,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10682,
    'CustomerID': 'ANTON',
    'OrderDate': '1997-09-25T00:00:00.000Z',
    'ShippedDate': '1997-10-01T00:00:00.000Z',
    'Freight': 36.13,
    'ShipName': 'Antonio Moreno Taquería',
    'ShipAddress': 'Mataderos 2312',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10683,
    'CustomerID': 'DUMON',
    'OrderDate': '1997-09-26T00:00:00.000Z',
    'ShippedDate': '1997-10-01T00:00:00.000Z',
    'Freight': 4.4,
    'ShipName': 'Du monde entier',
    'ShipAddress': '67, rue des Cinquante Otages',
    'ShipCity': 'Nantes',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10684,
    'CustomerID': 'OTTIK',
    'OrderDate': '1997-09-26T00:00:00.000Z',
    'ShippedDate': '1997-09-30T00:00:00.000Z',
    'Freight': 145.63,
    'ShipName': 'Ottilies Käseladen',
    'ShipAddress': 'Mehrheimerstr. 369',
    'ShipCity': 'Köln',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10685,
    'CustomerID': 'GOURL',
    'OrderDate': '1997-09-29T00:00:00.000Z',
    'ShippedDate': '1997-10-03T00:00:00.000Z',
    'Freight': 33.75,
    'ShipName': 'Gourmet Lanchonetes',
    'ShipAddress': 'Av. Brasil, 442',
    'ShipCity': 'Campinas',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10686,

```

```

    'CustomerID': 'PICCO',
    'OrderDate': '1997-09-30T00:00:00.000Z',
    'ShippedDate': '1997-10-08T00:00:00.000Z',
    'Freight': 96.5,
    'ShipName': 'Piccolo und mehr',
    'ShipAddress': 'Geislweg 14',
    'ShipCity': 'Salzburg',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10687,
    'CustomerID': 'HUNGO',
    'OrderDate': '1997-09-30T00:00:00.000Z',
    'ShippedDate': '1997-10-30T00:00:00.000Z',
    'Freight': 296.43,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10688,
    'CustomerID': 'VAFFE',
    'OrderDate': '1997-10-01T00:00:00.000Z',
    'ShippedDate': '1997-10-07T00:00:00.000Z',
    'Freight': 299.09,
    'ShipName': 'Vaffeljernet',
    'ShipAddress': 'Smagsloget 45',
    'ShipCity': 'Århus',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10689,
    'CustomerID': 'BERGS',
    'OrderDate': '1997-10-01T00:00:00.000Z',
    'ShippedDate': '1997-10-07T00:00:00.000Z',
    'Freight': 13.42,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10690,
    'CustomerID': 'HANAR',
    'OrderDate': '1997-10-02T00:00:00.000Z',
    'ShippedDate': '1997-10-03T00:00:00.000Z',
    'Freight': 15.8,
    'ShipName': 'Hanari Carnes',
    'ShipAddress': 'Rua do Paço, 67',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  }

```

```

    },
    {
        'OrderID': 10691,
        'CustomerID': 'QUICK',
        'OrderDate': '1997-10-03T00:00:00.000Z',
        'ShippedDate': '1997-10-22T00:00:00.000Z',
        'Freight': 810.05,
        'ShipName': 'QUICK-Stop',
        'ShipAddress': 'Taucherstraße 10',
        'ShipCity': 'Cunewalde',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10692,
        'CustomerID': 'ALFKI',
        'OrderDate': '1997-10-03T00:00:00.000Z',
        'ShippedDate': '1997-10-13T00:00:00.000Z',
        'Freight': 61.02,
        'ShipName': 'Alfred Futterkiste',
        'ShipAddress': 'Obere Str. 57',
        'ShipCity': 'Berlin',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10693,
        'CustomerID': 'WHITC',
        'OrderDate': '1997-10-06T00:00:00.000Z',
        'ShippedDate': '1997-10-10T00:00:00.000Z',
        'Freight': 139.34,
        'ShipName': 'White Clover Markets',
        'ShipAddress': '1029 - 12th Ave. S.',
        'ShipCity': 'Seattle',
        'ShipRegion': 'WA',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10694,
        'CustomerID': 'QUICK',
        'OrderDate': '1997-10-06T00:00:00.000Z',
        'ShippedDate': '1997-10-09T00:00:00.000Z',
        'Freight': 398.36,
        'ShipName': 'QUICK-Stop',
        'ShipAddress': 'Taucherstraße 10',
        'ShipCity': 'Cunewalde',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10695,
        'CustomerID': 'WILMK',
        'OrderDate': '1997-10-07T00:00:00.000Z',
        'ShippedDate': '1997-10-14T00:00:00.000Z',
        'Freight': 16.72,
        'ShipName': 'Wilman Kala',
        'ShipAddress': 'Keskuskatu 45',
    }

```

```

    'ShipCity': 'Helsinki',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10696,
    'CustomerID': 'WHITC',
    'OrderDate': '1997-10-08T00:00:00.000Z',
    'ShippedDate': '1997-10-14T00:00:00.000Z',
    'Freight': 102.55,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10697,
    'CustomerID': 'LINOD',
    'OrderDate': '1997-10-08T00:00:00.000Z',
    'ShippedDate': '1997-10-14T00:00:00.000Z',
    'Freight': 45.52,
    'ShipName': 'LINO-Delicatesses',
    'ShipAddress': 'Ave. 5 de Mayo Porlamar',
    'ShipCity': 'I. de Margarita',
    'ShipRegion': 'Nueva Esparta',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10698,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-10-09T00:00:00.000Z',
    'ShippedDate': '1997-10-17T00:00:00.000Z',
    'Freight': 272.47,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10699,
    'CustomerID': 'MORGK',
    'OrderDate': '1997-10-09T00:00:00.000Z',
    'ShippedDate': '1997-10-13T00:00:00.000Z',
    'Freight': 0.58,
    'ShipName': 'Morgenstern Gesundkost',
    'ShipAddress': 'Heerstr. 22',
    'ShipCity': 'Leipzig',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10700,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-10-10T00:00:00.000Z',
    'ShippedDate': '1997-10-16T00:00:00.000Z',

```

```

    'Freight': 65.1,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10701,
    'CustomerID': 'HUNGO',
    'OrderDate': '1997-10-13T00:00:00.000Z',
    'ShippedDate': '1997-10-15T00:00:00.000Z',
    'Freight': 220.31,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10702,
    'CustomerID': 'ALFKI',
    'OrderDate': '1997-10-13T00:00:00.000Z',
    'ShippedDate': '1997-10-21T00:00:00.000Z',
    'Freight': 23.94,
    'ShipName': 'Alfred Futterkiste',
    'ShipAddress': 'Obere Str. 57',
    'ShipCity': 'Berlin',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10703,
    'CustomerID': 'FOLKO',
    'OrderDate': '1997-10-14T00:00:00.000Z',
    'ShippedDate': '1997-10-20T00:00:00.000Z',
    'Freight': 152.3,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10704,
    'CustomerID': 'QUEEN',
    'OrderDate': '1997-10-14T00:00:00.000Z',
    'ShippedDate': '1997-11-07T00:00:00.000Z',
    'Freight': 4.78,
    'ShipName': 'Queen Cozinha',
    'ShipAddress': 'Alameda dos Canários, 891',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10705,

```

```

    'CustomerID': 'HILAA',
    'OrderDate': '1997-10-15T00:00:00.000Z',
    'ShippedDate': '1997-11-18T00:00:00.000Z',
    'Freight': 3.52,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10706,
    'CustomerID': 'OLDWO',
    'OrderDate': '1997-10-16T00:00:00.000Z',
    'ShippedDate': '1997-10-21T00:00:00.000Z',
    'Freight': 135.63,
    'ShipName': 'Old World Delicatessen',
    'ShipAddress': '2743 Bering St.',
    'ShipCity': 'Anchorage',
    'ShipRegion': 'AK',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10707,
    'CustomerID': 'AROUT',
    'OrderDate': '1997-10-16T00:00:00.000Z',
    'ShippedDate': '1997-10-23T00:00:00.000Z',
    'Freight': 21.74,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10708,
    'CustomerID': 'THEBI',
    'OrderDate': '1997-10-17T00:00:00.000Z',
    'ShippedDate': '1997-11-05T00:00:00.000Z',
    'Freight': 2.96,
    'ShipName': 'The Big Cheese',
    'ShipAddress': '89 Jefferson Way Suite 2',
    'ShipCity': 'Portland',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10709,
    'CustomerID': 'GOURL',
    'OrderDate': '1997-10-17T00:00:00.000Z',
    'ShippedDate': '1997-11-20T00:00:00.000Z',
    'Freight': 210.8,
    'ShipName': 'Gourmet Lanchonetes',
    'ShipAddress': 'Av. Brasil, 442',
    'ShipCity': 'Campinas',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  }

```

```

},
{
  'OrderID': 10710,
  'CustomerID': 'FRANS',
  'OrderDate': '1997-10-20T00:00:00.000Z',
  'ShippedDate': '1997-10-23T00:00:00.000Z',
  'Freight': 4.98,
  'ShipName': 'Franchi S.p.A.',
  'ShipAddress': 'Via Monte Bianco 34',
  'ShipCity': 'Torino',
  'ShipRegion': null,
  'ShipCountry': 'Italy'
},
{
  'OrderID': 10711,
  'CustomerID': 'SAVEA',
  'OrderDate': '1997-10-21T00:00:00.000Z',
  'ShippedDate': '1997-10-29T00:00:00.000Z',
  'Freight': 52.41,
  'ShipName': 'Save-a-lot Markets',
  'ShipAddress': '187 Suffolk Ln.',
  'ShipCity': 'Boise',
  'ShipRegion': 'ID',
  'ShipCountry': 'USA'
},
{
  'OrderID': 10712,
  'CustomerID': 'HUNGO',
  'OrderDate': '1997-10-21T00:00:00.000Z',
  'ShippedDate': '1997-10-31T00:00:00.000Z',
  'Freight': 89.93,
  'ShipName': 'Hungry Owl All-Night Grocers',
  'ShipAddress': '8 Johnstown Road',
  'ShipCity': 'Cork',
  'ShipRegion': 'Co. Cork',
  'ShipCountry': 'Ireland'
},
{
  'OrderID': 10713,
  'CustomerID': 'SAVEA',
  'OrderDate': '1997-10-22T00:00:00.000Z',
  'ShippedDate': '1997-10-24T00:00:00.000Z',
  'Freight': 167.05,
  'ShipName': 'Save-a-lot Markets',
  'ShipAddress': '187 Suffolk Ln.',
  'ShipCity': 'Boise',
  'ShipRegion': 'ID',
  'ShipCountry': 'USA'
},
{
  'OrderID': 10714,
  'CustomerID': 'SAVEA',
  'OrderDate': '1997-10-22T00:00:00.000Z',
  'ShippedDate': '1997-10-27T00:00:00.000Z',
  'Freight': 24.49,
  'ShipName': 'Save-a-lot Markets',
  'ShipAddress': '187 Suffolk Ln.',

```

```

        'ShipCity': 'Boise',
        'ShipRegion': 'ID',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10715,
        'CustomerID': 'BONAP',
        'OrderDate': '1997-10-23T00:00:00.000Z',
        'ShippedDate': '1997-10-29T00:00:00.000Z',
        'Freight': 63.2,
        'ShipName': 'Bon app',
        'ShipAddress': '12, rue des Bouchers',
        'ShipCity': 'Marseille',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10716,
        'CustomerID': 'RANCH',
        'OrderDate': '1997-10-24T00:00:00.000Z',
        'ShippedDate': '1997-10-27T00:00:00.000Z',
        'Freight': 22.57,
        'ShipName': 'Rancho grande',
        'ShipAddress': 'Av. del Libertador 900',
        'ShipCity': 'Buenos Aires',
        'ShipRegion': null,
        'ShipCountry': 'Argentina'
    },
    {
        'OrderID': 10717,
        'CustomerID': 'FRANK',
        'OrderDate': '1997-10-24T00:00:00.000Z',
        'ShippedDate': '1997-10-29T00:00:00.000Z',
        'Freight': 59.25,
        'ShipName': 'Frankenversand',
        'ShipAddress': 'Berliner Platz 43',
        'ShipCity': 'München',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10718,
        'CustomerID': 'KOENE',
        'OrderDate': '1997-10-27T00:00:00.000Z',
        'ShippedDate': '1997-10-29T00:00:00.000Z',
        'Freight': 170.88,
        'ShipName': 'Königlich Essen',
        'ShipAddress': 'Maubelstr. 90',
        'ShipCity': 'Brandenburg',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10719,
        'CustomerID': 'LETSS',
        'OrderDate': '1997-10-27T00:00:00.000Z',
        'ShippedDate': '1997-11-05T00:00:00.000Z',

```



```

    'Freight': 51.44,
    'ShipName': 'Let\' Stop N Shop',
    'ShipAddress': '87 Polk St. Suite 5',
    'ShipCity': 'San Francisco',
    'ShipRegion': 'CA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10720,
    'CustomerID': 'QUEDE',
    'OrderDate': '1997-10-28T00:00:00.000Z',
    'ShippedDate': '1997-11-05T00:00:00.000Z',
    'Freight': 9.53,
    'ShipName': 'Que Delícia',
    'ShipAddress': 'Rua da Panificadora, 12',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10721,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-10-29T00:00:00.000Z',
    'ShippedDate': '1997-10-31T00:00:00.000Z',
    'Freight': 48.92,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10722,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-10-29T00:00:00.000Z',
    'ShippedDate': '1997-11-04T00:00:00.000Z',
    'Freight': 74.58,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10723,
    'CustomerID': 'WHITC',
    'OrderDate': '1997-10-30T00:00:00.000Z',
    'ShippedDate': '1997-11-25T00:00:00.000Z',
    'Freight': 21.72,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10724,

```

```

    'CustomerID': 'MEREP',
    'OrderDate': '1997-10-30T00:00:00.000Z',
    'ShippedDate': '1997-11-05T00:00:00.000Z',
    'Freight': 57.75,
    'ShipName': 'Mère Paillarde',
    'ShipAddress': '43 rue St. Laurent',
    'ShipCity': 'Montréal',
    'ShipRegion': 'Québec',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10725,
    'CustomerID': 'FAMIA',
    'OrderDate': '1997-10-31T00:00:00.000Z',
    'ShippedDate': '1997-11-05T00:00:00.000Z',
    'Freight': 10.83,
    'ShipName': 'Familia Arquibaldo',
    'ShipAddress': 'Rua Orós, 92',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10726,
    'CustomerID': 'EASTC',
    'OrderDate': '1997-11-03T00:00:00.000Z',
    'ShippedDate': '1997-12-05T00:00:00.000Z',
    'Freight': 16.56,
    'ShipName': 'Eastern Connection',
    'ShipAddress': '35 King George',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10727,
    'CustomerID': 'REGGC',
    'OrderDate': '1997-11-03T00:00:00.000Z',
    'ShippedDate': '1997-12-05T00:00:00.000Z',
    'Freight': 89.9,
    'ShipName': 'Reggiani Caseifici',
    'ShipAddress': 'Strada Provinciale 124',
    'ShipCity': 'Reggio Emilia',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10728,
    'CustomerID': 'QUEEN',
    'OrderDate': '1997-11-04T00:00:00.000Z',
    'ShippedDate': '1997-11-11T00:00:00.000Z',
    'Freight': 58.33,
    'ShipName': 'Queen Cozinha',
    'ShipAddress': 'Alameda dos Canários, 891',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  }

```

```

    },
    {
        'OrderID': 10729,
        'CustomerID': 'LINOD',
        'OrderDate': '1997-11-04T00:00:00.000Z',
        'ShippedDate': '1997-11-14T00:00:00.000Z',
        'Freight': 141.06,
        'ShipName': 'LINO-Delicateses',
        'ShipAddress': 'Ave. 5 de Mayo Porlamar',
        'ShipCity': 'I. de Margarita',
        'ShipRegion': 'Nueva Esparta',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10730,
        'CustomerID': 'BONAP',
        'OrderDate': '1997-11-05T00:00:00.000Z',
        'ShippedDate': '1997-11-14T00:00:00.000Z',
        'Freight': 20.12,
        'ShipName': 'Bon app',
        'ShipAddress': '12, rue des Bouchers',
        'ShipCity': 'Marseille',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10731,
        'CustomerID': 'CHOPS',
        'OrderDate': '1997-11-06T00:00:00.000Z',
        'ShippedDate': '1997-11-14T00:00:00.000Z',
        'Freight': 96.65,
        'ShipName': 'Chop-suey Chinese',
        'ShipAddress': 'Hauptstr. 31',
        'ShipCity': 'Bern',
        'ShipRegion': null,
        'ShipCountry': 'Switzerland'
    },
    {
        'OrderID': 10732,
        'CustomerID': 'BONAP',
        'OrderDate': '1997-11-06T00:00:00.000Z',
        'ShippedDate': '1997-11-07T00:00:00.000Z',
        'Freight': 16.97,
        'ShipName': 'Bon app',
        'ShipAddress': '12, rue des Bouchers',
        'ShipCity': 'Marseille',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10733,
        'CustomerID': 'BERGS',
        'OrderDate': '1997-11-07T00:00:00.000Z',
        'ShippedDate': '1997-11-10T00:00:00.000Z',
        'Freight': 110.11,
        'ShipName': 'Berglunds snabbköp',
        'ShipAddress': 'Berguvsvägen 8',
    }

```

```

        'ShipCity': 'Luleå',
        'ShipRegion': null,
        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 10734,
        'CustomerID': 'GOURL',
        'OrderDate': '1997-11-07T00:00:00.000Z',
        'ShippedDate': '1997-11-12T00:00:00.000Z',
        'Freight': 1.63,
        'ShipName': 'Gourmet Lanchonetes',
        'ShipAddress': 'Av. Brasil, 442',
        'ShipCity': 'Campinas',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10735,
        'CustomerID': 'LETSS',
        'OrderDate': '1997-11-10T00:00:00.000Z',
        'ShippedDate': '1997-11-21T00:00:00.000Z',
        'Freight': 45.97,
        'ShipName': 'Let\' Stop N Shop',
        'ShipAddress': '87 Polk St. Suite 5',
        'ShipCity': 'San Francisco',
        'ShipRegion': 'CA',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10736,
        'CustomerID': 'HUNGO',
        'OrderDate': '1997-11-11T00:00:00.000Z',
        'ShippedDate': '1997-11-21T00:00:00.000Z',
        'Freight': 44.1,
        'ShipName': 'Hungry Owl All-Night Grocers',
        'ShipAddress': '8 Johnstown Road',
        'ShipCity': 'Cork',
        'ShipRegion': 'Co. Cork',
        'ShipCountry': 'Ireland'
    },
    {
        'OrderID': 10737,
        'CustomerID': 'VINET',
        'OrderDate': '1997-11-11T00:00:00.000Z',
        'ShippedDate': '1997-11-18T00:00:00.000Z',
        'Freight': 7.79,
        'ShipName': 'Vins et alcools Chevalier',
        'ShipAddress': '59 rue de l\'Abbaye',
        'ShipCity': 'Reims',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10738,
        'CustomerID': 'SPECB',
        'OrderDate': '1997-11-12T00:00:00.000Z',
        'ShippedDate': '1997-11-18T00:00:00.000Z',

```

```

    'Freight': 2.91,
    'ShipName': 'Spécialités du monde',
    'ShipAddress': '25, rue Lauriston',
    'ShipCity': 'Paris',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10739,
    'CustomerID': 'VINET',
    'OrderDate': '1997-11-12T00:00:00.000Z',
    'ShippedDate': '1997-11-17T00:00:00.000Z',
    'Freight': 11.08,
    'ShipName': 'Vins et alcools Chevalier',
    'ShipAddress': '59 rue de l'Abbaye',
    'ShipCity': 'Reims',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10740,
    'CustomerID': 'WHITC',
    'OrderDate': '1997-11-13T00:00:00.000Z',
    'ShippedDate': '1997-11-25T00:00:00.000Z',
    'Freight': 81.88,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10741,
    'CustomerID': 'AROUT',
    'OrderDate': '1997-11-14T00:00:00.000Z',
    'ShippedDate': '1997-11-18T00:00:00.000Z',
    'Freight': 10.96,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10742,
    'CustomerID': 'BOTTM',
    'OrderDate': '1997-11-14T00:00:00.000Z',
    'ShippedDate': '1997-11-18T00:00:00.000Z',
    'Freight': 243.73,
    'ShipName': 'Bottom-Dollar Markets',
    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10743,

```

```

    'CustomerID': 'AROUT',
    'OrderDate': '1997-11-17T00:00:00.000Z',
    'ShippedDate': '1997-11-21T00:00:00.000Z',
    'Freight': 23.72,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10744,
    'CustomerID': 'VAFFE',
    'OrderDate': '1997-11-17T00:00:00.000Z',
    'ShippedDate': '1997-11-24T00:00:00.000Z',
    'Freight': 69.19,
    'ShipName': 'Vaffeljernet',
    'ShipAddress': 'Smagsloget 45',
    'ShipCity': 'Århus',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10745,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-11-18T00:00:00.000Z',
    'ShippedDate': '1997-11-27T00:00:00.000Z',
    'Freight': 3.52,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10746,
    'CustomerID': 'CHOPS',
    'OrderDate': '1997-11-19T00:00:00.000Z',
    'ShippedDate': '1997-11-21T00:00:00.000Z',
    'Freight': 31.43,
    'ShipName': 'Chop-suey Chinese',
    'ShipAddress': 'Hauptstr. 31',
    'ShipCity': 'Bern',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10747,
    'CustomerID': 'PICCO',
    'OrderDate': '1997-11-19T00:00:00.000Z',
    'ShippedDate': '1997-11-26T00:00:00.000Z',
    'Freight': 117.33,
    'ShipName': 'Piccolo und mehr',
    'ShipAddress': 'Geislweg 14',
    'ShipCity': 'Salzburg',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  }

```

```

},
{
  'OrderID': 10748,
  'CustomerID': 'SAVEA',
  'OrderDate': '1997-11-20T00:00:00.000Z',
  'ShippedDate': '1997-11-28T00:00:00.000Z',
  'Freight': 232.55,
  'ShipName': 'Save-a-lot Markets',
  'ShipAddress': '187 Suffolk Ln.',
  'ShipCity': 'Boise',
  'ShipRegion': 'ID',
  'ShipCountry': 'USA'
},
{
  'OrderID': 10749,
  'CustomerID': 'ISLAT',
  'OrderDate': '1997-11-20T00:00:00.000Z',
  'ShippedDate': '1997-12-19T00:00:00.000Z',
  'Freight': 61.53,
  'ShipName': 'Island Trading',
  'ShipAddress': 'Garden House Crowther Way',
  'ShipCity': 'Cowes',
  'ShipRegion': 'Isle of Wight',
  'ShipCountry': 'UK'
},
{
  'OrderID': 10750,
  'CustomerID': 'WARTH',
  'OrderDate': '1997-11-21T00:00:00.000Z',
  'ShippedDate': '1997-11-24T00:00:00.000Z',
  'Freight': 79.3,
  'ShipName': 'Wartian Herkku',
  'ShipAddress': 'Torikatu 38',
  'ShipCity': 'Oulu',
  'ShipRegion': null,
  'ShipCountry': 'Finland'
},
{
  'OrderID': 10751,
  'CustomerID': 'RICSU',
  'OrderDate': '1997-11-24T00:00:00.000Z',
  'ShippedDate': '1997-12-03T00:00:00.000Z',
  'Freight': 130.79,
  'ShipName': 'Richter Supermarkt',
  'ShipAddress': 'Starenweg 5',
  'ShipCity': 'Genève',
  'ShipRegion': null,
  'ShipCountry': 'Switzerland'
},
{
  'OrderID': 10752,
  'CustomerID': 'NORTS',
  'OrderDate': '1997-11-24T00:00:00.000Z',
  'ShippedDate': '1997-11-28T00:00:00.000Z',
  'Freight': 1.39,
  'ShipName': 'North/South',
  'ShipAddress': 'South House 300 Queensbridge',

```

```

        'ShipCity': 'London',
        'ShipRegion': null,
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10753,
        'CustomerID': 'FRANS',
        'OrderDate': '1997-11-25T00:00:00.000Z',
        'ShippedDate': '1997-11-27T00:00:00.000Z',
        'Freight': 7.7,
        'ShipName': 'Franchi S.p.A.',
        'ShipAddress': 'Via Monte Bianco 34',
        'ShipCity': 'Torino',
        'ShipRegion': null,
        'ShipCountry': 'Italy'
    },
    {
        'OrderID': 10754,
        'CustomerID': 'MAGAA',
        'OrderDate': '1997-11-25T00:00:00.000Z',
        'ShippedDate': '1997-11-27T00:00:00.000Z',
        'Freight': 2.38,
        'ShipName': 'Magazzini Alimentari Riuniti',
        'ShipAddress': 'Via Ludovico il Moro 22',
        'ShipCity': 'Bergamo',
        'ShipRegion': null,
        'ShipCountry': 'Italy'
    },
    {
        'OrderID': 10755,
        'CustomerID': 'BONAP',
        'OrderDate': '1997-11-26T00:00:00.000Z',
        'ShippedDate': '1997-11-28T00:00:00.000Z',
        'Freight': 16.71,
        'ShipName': 'Bon app',
        'ShipAddress': '12, rue des Bouchers',
        'ShipCity': 'Marseille',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10756,
        'CustomerID': 'SPLIR',
        'OrderDate': '1997-11-27T00:00:00.000Z',
        'ShippedDate': '1997-12-02T00:00:00.000Z',
        'Freight': 73.21,
        'ShipName': 'Split Rail Beer & Ale',
        'ShipAddress': 'P.O. Box 555',
        'ShipCity': 'Lander',
        'ShipRegion': 'WY',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10757,
        'CustomerID': 'SAVEA',
        'OrderDate': '1997-11-27T00:00:00.000Z',
        'ShippedDate': '1997-12-15T00:00:00.000Z',

```



```

    'Freight': 8.19,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10758,
    'CustomerID': 'RICSU',
    'OrderDate': '1997-11-28T00:00:00.000Z',
    'ShippedDate': '1997-12-04T00:00:00.000Z',
    'Freight': 138.17,
    'ShipName': 'Richter Supermarkt',
    'ShipAddress': 'Starenweg 5',
    'ShipCity': 'Genève',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10759,
    'CustomerID': 'ANATR',
    'OrderDate': '1997-11-28T00:00:00.000Z',
    'ShippedDate': '1997-12-12T00:00:00.000Z',
    'Freight': 11.99,
    'ShipName': 'Ana Trujillo Emparedados y helados',
    'ShipAddress': 'Avda. de la Constitución 2222',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10760,
    'CustomerID': 'MAISD',
    'OrderDate': '1997-12-01T00:00:00.000Z',
    'ShippedDate': '1997-12-10T00:00:00.000Z',
    'Freight': 155.64,
    'ShipName': 'Maison Dewey',
    'ShipAddress': 'Rue Joseph-Bens 532',
    'ShipCity': 'Bruxelles',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10761,
    'CustomerID': 'RATTC',
    'OrderDate': '1997-12-02T00:00:00.000Z',
    'ShippedDate': '1997-12-08T00:00:00.000Z',
    'Freight': 18.66,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10762,

```

```

    'CustomerID': 'FOLKO',
    'OrderDate': '1997-12-02T00:00:00.000Z',
    'ShippedDate': '1997-12-09T00:00:00.000Z',
    'Freight': 328.74,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10763,
    'CustomerID': 'FOLIG',
    'OrderDate': '1997-12-03T00:00:00.000Z',
    'ShippedDate': '1997-12-08T00:00:00.000Z',
    'Freight': 37.35,
    'ShipName': 'Folies gourmandes',
    'ShipAddress': '184, chaussée de Tournai',
    'ShipCity': 'Lille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10764,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-12-03T00:00:00.000Z',
    'ShippedDate': '1997-12-08T00:00:00.000Z',
    'Freight': 145.45,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10765,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-12-04T00:00:00.000Z',
    'ShippedDate': '1997-12-09T00:00:00.000Z',
    'Freight': 42.74,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10766,
    'CustomerID': 'OTTIK',
    'OrderDate': '1997-12-05T00:00:00.000Z',
    'ShippedDate': '1997-12-09T00:00:00.000Z',
    'Freight': 157.55,
    'ShipName': 'Ottilies Käseladen',
    'ShipAddress': 'Mehrheimerstr. 369',
    'ShipCity': 'Köln',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  }

```

```

},
{
  'OrderID': 10767,
  'CustomerID': 'SUPRD',
  'OrderDate': '1997-12-05T00:00:00.000Z',
  'ShippedDate': '1997-12-15T00:00:00.000Z',
  'Freight': 1.59,
  'ShipName': 'Suprêmes délices',
  'ShipAddress': 'Boulevard Tirou, 255',
  'ShipCity': 'Charleroi',
  'ShipRegion': null,
  'ShipCountry': 'Belgium'
},
{
  'OrderID': 10768,
  'CustomerID': 'AROUT',
  'OrderDate': '1997-12-08T00:00:00.000Z',
  'ShippedDate': '1997-12-15T00:00:00.000Z',
  'Freight': 146.32,
  'ShipName': 'Around the Horn',
  'ShipAddress': 'Brook Farm Stratford St. Mary',
  'ShipCity': 'Colchester',
  'ShipRegion': 'Essex',
  'ShipCountry': 'UK'
},
{
  'OrderID': 10769,
  'CustomerID': 'VAFFE',
  'OrderDate': '1997-12-08T00:00:00.000Z',
  'ShippedDate': '1997-12-12T00:00:00.000Z',
  'Freight': 65.06,
  'ShipName': 'Vaffeljernet',
  'ShipAddress': 'Smagsloget 45',
  'ShipCity': 'Århus',
  'ShipRegion': null,
  'ShipCountry': 'Denmark'
},
{
  'OrderID': 10770,
  'CustomerID': 'HANAR',
  'OrderDate': '1997-12-09T00:00:00.000Z',
  'ShippedDate': '1997-12-17T00:00:00.000Z',
  'Freight': 5.32,
  'ShipName': 'Hanari Carnes',
  'ShipAddress': 'Rua do Paço, 67',
  'ShipCity': 'Rio de Janeiro',
  'ShipRegion': 'RJ',
  'ShipCountry': 'Brazil'
},
{
  'OrderID': 10771,
  'CustomerID': 'ERNSH',
  'OrderDate': '1997-12-10T00:00:00.000Z',
  'ShippedDate': '1998-01-02T00:00:00.000Z',
  'Freight': 11.19,
  'ShipName': 'Ernst Handel',
  'ShipAddress': 'Kirchgasse 6',

```

```

    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10772,
    'CustomerID': 'LEHMS',
    'OrderDate': '1997-12-10T00:00:00.000Z',
    'ShippedDate': '1997-12-19T00:00:00.000Z',
    'Freight': 91.28,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10773,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-12-11T00:00:00.000Z',
    'ShippedDate': '1997-12-16T00:00:00.000Z',
    'Freight': 96.43,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10774,
    'CustomerID': 'FOLKO',
    'OrderDate': '1997-12-11T00:00:00.000Z',
    'ShippedDate': '1997-12-12T00:00:00.000Z',
    'Freight': 48.2,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10775,
    'CustomerID': 'THECR',
    'OrderDate': '1997-12-12T00:00:00.000Z',
    'ShippedDate': '1997-12-26T00:00:00.000Z',
    'Freight': 20.25,
    'ShipName': 'The Cracker Box',
    'ShipAddress': '55 Grizzly Peak Rd.',
    'ShipCity': 'Butte',
    'ShipRegion': 'MT',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10776,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-12-15T00:00:00.000Z',
    'ShippedDate': '1997-12-18T00:00:00.000Z',

```

```

    'Freight': 351.53,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10777,
    'CustomerID': 'GOURL',
    'OrderDate': '1997-12-15T00:00:00.000Z',
    'ShippedDate': '1998-01-21T00:00:00.000Z',
    'Freight': 3.01,
    'ShipName': 'Gourmet Lanchonetes',
    'ShipAddress': 'Av. Brasil, 442',
    'ShipCity': 'Campinas',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10778,
    'CustomerID': 'BERGS',
    'OrderDate': '1997-12-16T00:00:00.000Z',
    'ShippedDate': '1997-12-24T00:00:00.000Z',
    'Freight': 6.79,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10779,
    'CustomerID': 'MORGK',
    'OrderDate': '1997-12-16T00:00:00.000Z',
    'ShippedDate': '1998-01-14T00:00:00.000Z',
    'Freight': 58.13,
    'ShipName': 'Morgenstern Gesundkost',
    'ShipAddress': 'Heerstr. 22',
    'ShipCity': 'Leipzig',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10780,
    'CustomerID': 'LILAS',
    'OrderDate': '1997-12-16T00:00:00.000Z',
    'ShippedDate': '1997-12-25T00:00:00.000Z',
    'Freight': 42.13,
    'ShipName': 'LILA-Supermercado',
    'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10781,

```

```

      'CustomerID': 'WARTH',
      'OrderDate': '1997-12-17T00:00:00.000Z',
      'ShippedDate': '1997-12-19T00:00:00.000Z',
      'Freight': 73.16,
      'ShipName': 'Wartian Herkku',
      'ShipAddress': 'Torikatu 38',
      'ShipCity': 'Oulu',
      'ShipRegion': null,
      'ShipCountry': 'Finland'
    },
    {
      'OrderID': 10782,
      'CustomerID': 'CACTU',
      'OrderDate': '1997-12-17T00:00:00.000Z',
      'ShippedDate': '1997-12-22T00:00:00.000Z',
      'Freight': 1.1,
      'ShipName': 'Cactus Comidas para llevar',
      'ShipAddress': 'Cerrito 333',
      'ShipCity': 'Buenos Aires',
      'ShipRegion': null,
      'ShipCountry': 'Argentina'
    },
    {
      'OrderID': 10783,
      'CustomerID': 'HANAR',
      'OrderDate': '1997-12-18T00:00:00.000Z',
      'ShippedDate': '1997-12-19T00:00:00.000Z',
      'Freight': 124.98,
      'ShipName': 'Hanari Carnes',
      'ShipAddress': 'Rua do Paço, 67',
      'ShipCity': 'Rio de Janeiro',
      'ShipRegion': 'RJ',
      'ShipCountry': 'Brazil'
    },
    {
      'OrderID': 10784,
      'CustomerID': 'MAGAA',
      'OrderDate': '1997-12-18T00:00:00.000Z',
      'ShippedDate': '1997-12-22T00:00:00.000Z',
      'Freight': 70.09,
      'ShipName': 'Magazzini Alimentari Riuniti',
      'ShipAddress': 'Via Ludovico il Moro 22',
      'ShipCity': 'Bergamo',
      'ShipRegion': null,
      'ShipCountry': 'Italy'
    },
    {
      'OrderID': 10785,
      'CustomerID': 'GROSR',
      'OrderDate': '1997-12-18T00:00:00.000Z',
      'ShippedDate': '1997-12-24T00:00:00.000Z',
      'Freight': 1.51,
      'ShipName': 'GROSELLA-Restaurante',
      'ShipAddress': '5ª Ave. Los Palos Grandes',
      'ShipCity': 'Caracas',
      'ShipRegion': 'DF',
      'ShipCountry': 'Venezuela'
    }
  ]

```

```

    },
    {
        'OrderID': 10786,
        'CustomerID': 'QUEEN',
        'OrderDate': '1997-12-19T00:00:00.000Z',
        'ShippedDate': '1997-12-23T00:00:00.000Z',
        'Freight': 110.87,
        'ShipName': 'Queen Cozinha',
        'ShipAddress': 'Alameda dos Canários, 891',
        'ShipCity': 'Sao Paulo',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10787,
        'CustomerID': 'LAMAI',
        'OrderDate': '1997-12-19T00:00:00.000Z',
        'ShippedDate': '1997-12-26T00:00:00.000Z',
        'Freight': 249.93,
        'ShipName': 'La maison d\'Asie',
        'ShipAddress': '1 rue Alsace-Lorraine',
        'ShipCity': 'Toulouse',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10788,
        'CustomerID': 'QUICK',
        'OrderDate': '1997-12-22T00:00:00.000Z',
        'ShippedDate': '1998-01-19T00:00:00.000Z',
        'Freight': 42.7,
        'ShipName': 'QUICK-Stop',
        'ShipAddress': 'Taucherstraße 10',
        'ShipCity': 'Cunewalde',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10789,
        'CustomerID': 'FOLIG',
        'OrderDate': '1997-12-22T00:00:00.000Z',
        'ShippedDate': '1997-12-31T00:00:00.000Z',
        'Freight': 100.6,
        'ShipName': 'Folies gourmandes',
        'ShipAddress': '184, chaussée de Tournai',
        'ShipCity': 'Lille',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10790,
        'CustomerID': 'GOURL',
        'OrderDate': '1997-12-22T00:00:00.000Z',
        'ShippedDate': '1997-12-26T00:00:00.000Z',
        'Freight': 28.23,
        'ShipName': 'Gourmet Lanchonetes',
        'ShipAddress': 'Av. Brasil, 442',
    }

```

```

    'ShipCity': 'Campinas',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10791,
    'CustomerID': 'FRANK',
    'OrderDate': '1997-12-23T00:00:00.000Z',
    'ShippedDate': '1998-01-01T00:00:00.000Z',
    'Freight': 16.85,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10792,
    'CustomerID': 'WOLZA',
    'OrderDate': '1997-12-23T00:00:00.000Z',
    'ShippedDate': '1997-12-31T00:00:00.000Z',
    'Freight': 23.79,
    'ShipName': 'Wolski Zajazd',
    'ShipAddress': 'ul. Filtrowa 68',
    'ShipCity': 'Warszawa',
    'ShipRegion': null,
    'ShipCountry': 'Poland'
  },
  {
    'OrderID': 10793,
    'CustomerID': 'AROUT',
    'OrderDate': '1997-12-24T00:00:00.000Z',
    'ShippedDate': '1998-01-08T00:00:00.000Z',
    'Freight': 4.52,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10794,
    'CustomerID': 'QUEDE',
    'OrderDate': '1997-12-24T00:00:00.000Z',
    'ShippedDate': '1998-01-02T00:00:00.000Z',
    'Freight': 21.49,
    'ShipName': 'Que Delícia',
    'ShipAddress': 'Rua da Panificadora, 12',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10795,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-12-24T00:00:00.000Z',
    'ShippedDate': '1998-01-20T00:00:00.000Z',

```



```

    'Freight': 126.66,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10796,
    'CustomerID': 'HILAA',
    'OrderDate': '1997-12-25T00:00:00.000Z',
    'ShippedDate': '1998-01-14T00:00:00.000Z',
    'Freight': 26.52,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10797,
    'CustomerID': 'DRACD',
    'OrderDate': '1997-12-25T00:00:00.000Z',
    'ShippedDate': '1998-01-05T00:00:00.000Z',
    'Freight': 33.35,
    'ShipName': 'Drachenblut Delikatessen',
    'ShipAddress': 'Walserweg 21',
    'ShipCity': 'Aachen',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10798,
    'CustomerID': 'ISLAT',
    'OrderDate': '1997-12-26T00:00:00.000Z',
    'ShippedDate': '1998-01-05T00:00:00.000Z',
    'Freight': 2.33,
    'ShipName': 'Island Trading',
    'ShipAddress': 'Garden House Crowther Way',
    'ShipCity': 'Cowes',
    'ShipRegion': 'Isle of Wight',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10799,
    'CustomerID': 'KOENE',
    'OrderDate': '1997-12-26T00:00:00.000Z',
    'ShippedDate': '1998-01-05T00:00:00.000Z',
    'Freight': 30.76,
    'ShipName': 'Königlich Essen',
    'ShipAddress': 'Maubelstr. 90',
    'ShipCity': 'Brandenburg',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10800,

```

```

      'CustomerID': 'SEVES',
      'OrderDate': '1997-12-26T00:00:00.000Z',
      'ShippedDate': '1998-01-05T00:00:00.000Z',
      'Freight': 137.44,
      'ShipName': 'Seven Seas Imports',
      'ShipAddress': '90 Wadhurst Rd.',
      'ShipCity': 'London',
      'ShipRegion': null,
      'ShipCountry': 'UK'
    },
    {
      'OrderID': 10801,
      'CustomerID': 'BOLID',
      'OrderDate': '1997-12-29T00:00:00.000Z',
      'ShippedDate': '1997-12-31T00:00:00.000Z',
      'Freight': 97.09,
      'ShipName': 'Bólido Comidas preparadas',
      'ShipAddress': 'C/ Araquil, 67',
      'ShipCity': 'Madrid',
      'ShipRegion': null,
      'ShipCountry': 'Spain'
    },
    {
      'OrderID': 10802,
      'CustomerID': 'SIMOB',
      'OrderDate': '1997-12-29T00:00:00.000Z',
      'ShippedDate': '1998-01-02T00:00:00.000Z',
      'Freight': 257.26,
      'ShipName': 'Simons bistro',
      'ShipAddress': 'Vinbæltet 34',
      'ShipCity': 'Kobenhavn',
      'ShipRegion': null,
      'ShipCountry': 'Denmark'
    },
    {
      'OrderID': 10803,
      'CustomerID': 'WELLI',
      'OrderDate': '1997-12-30T00:00:00.000Z',
      'ShippedDate': '1998-01-06T00:00:00.000Z',
      'Freight': 55.23,
      'ShipName': 'Wellington Importadora',
      'ShipAddress': 'Rua do Mercado, 12',
      'ShipCity': 'Resende',
      'ShipRegion': 'SP',
      'ShipCountry': 'Brazil'
    },
    {
      'OrderID': 10804,
      'CustomerID': 'SEVES',
      'OrderDate': '1997-12-30T00:00:00.000Z',
      'ShippedDate': '1998-01-07T00:00:00.000Z',
      'Freight': 27.33,
      'ShipName': 'Seven Seas Imports',
      'ShipAddress': '90 Wadhurst Rd.',
      'ShipCity': 'London',
      'ShipRegion': null,
      'ShipCountry': 'UK'
    }
  ]

```

```

},
{
  'OrderID': 10805,
  'CustomerID': 'THEBI',
  'OrderDate': '1997-12-30T00:00:00.000Z',
  'ShippedDate': '1998-01-09T00:00:00.000Z',
  'Freight': 237.34,
  'ShipName': 'The Big Cheese',
  'ShipAddress': '89 Jefferson Way Suite 2',
  'ShipCity': 'Portland',
  'ShipRegion': 'OR',
  'ShipCountry': 'USA'
},
{
  'OrderID': 10806,
  'CustomerID': 'VICTE',
  'OrderDate': '1997-12-31T00:00:00.000Z',
  'ShippedDate': '1998-01-05T00:00:00.000Z',
  'Freight': 22.11,
  'ShipName': 'Victuailles en stock',
  'ShipAddress': '2, rue du Commerce',
  'ShipCity': 'Lyon',
  'ShipRegion': null,
  'ShipCountry': 'France'
},
{
  'OrderID': 10807,
  'CustomerID': 'FRANS',
  'OrderDate': '1997-12-31T00:00:00.000Z',
  'ShippedDate': '1998-01-30T00:00:00.000Z',
  'Freight': 1.36,
  'ShipName': 'Franchi S.p.A.',
  'ShipAddress': 'Via Monte Bianco 34',
  'ShipCity': 'Torino',
  'ShipRegion': null,
  'ShipCountry': 'Italy'
},
{
  'OrderID': 10808,
  'CustomerID': 'OLDWO',
  'OrderDate': '1998-01-01T00:00:00.000Z',
  'ShippedDate': '1998-01-09T00:00:00.000Z',
  'Freight': 45.53,
  'ShipName': 'Old World Delicatessen',
  'ShipAddress': '2743 Bering St.',
  'ShipCity': 'Anchorage',
  'ShipRegion': 'AK',
  'ShipCountry': 'USA'
},
{
  'OrderID': 10809,
  'CustomerID': 'WELLI',
  'OrderDate': '1998-01-01T00:00:00.000Z',
  'ShippedDate': '1998-01-07T00:00:00.000Z',
  'Freight': 4.87,
  'ShipName': 'Wellington Importadora',
  'ShipAddress': 'Rua do Mercado, 12',

```

```

        'ShipCity': 'Resende',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10810,
        'CustomerID': 'LAUGB',
        'OrderDate': '1998-01-01T00:00:00.000Z',
        'ShippedDate': '1998-01-07T00:00:00.000Z',
        'Freight': 4.33,
        'ShipName': 'Laughing Bacchus Wine Cellars',
        'ShipAddress': '2319 Elm St.',
        'ShipCity': 'Vancouver',
        'ShipRegion': 'BC',
        'ShipCountry': 'Canada'
    },
    {
        'OrderID': 10811,
        'CustomerID': 'LINOD',
        'OrderDate': '1998-01-02T00:00:00.000Z',
        'ShippedDate': '1998-01-08T00:00:00.000Z',
        'Freight': 31.22,
        'ShipName': 'LINO-Delicatesses',
        'ShipAddress': 'Ave. 5 de Mayo Porlamar',
        'ShipCity': 'I. de Margarita',
        'ShipRegion': 'Nueva Esparta',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10812,
        'CustomerID': 'REGGC',
        'OrderDate': '1998-01-02T00:00:00.000Z',
        'ShippedDate': '1998-01-12T00:00:00.000Z',
        'Freight': 59.78,
        'ShipName': 'Reggiani Caseifici',
        'ShipAddress': 'Strada Provinciale 124',
        'ShipCity': 'Reggio Emilia',
        'ShipRegion': null,
        'ShipCountry': 'Italy'
    },
    {
        'OrderID': 10813,
        'CustomerID': 'RICAR',
        'OrderDate': '1998-01-05T00:00:00.000Z',
        'ShippedDate': '1998-01-09T00:00:00.000Z',
        'Freight': 47.38,
        'ShipName': 'Ricardo Adocicados',
        'ShipAddress': 'Av. Copacabana, 267',
        'ShipCity': 'Rio de Janeiro',
        'ShipRegion': 'RJ',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10814,
        'CustomerID': 'VICTE',
        'OrderDate': '1998-01-05T00:00:00.000Z',
        'ShippedDate': '1998-01-14T00:00:00.000Z',

```

```

    'Freight': 130.94,
    'ShipName': 'Victuailles en stock',
    'ShipAddress': '2, rue du Commerce',
    'ShipCity': 'Lyon',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10815,
    'CustomerID': 'SAVEA',
    'OrderDate': '1998-01-05T00:00:00.000Z',
    'ShippedDate': '1998-01-14T00:00:00.000Z',
    'Freight': 14.62,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10816,
    'CustomerID': 'GREAL',
    'OrderDate': '1998-01-06T00:00:00.000Z',
    'ShippedDate': '1998-02-04T00:00:00.000Z',
    'Freight': 719.78,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10817,
    'CustomerID': 'KOENE',
    'OrderDate': '1998-01-06T00:00:00.000Z',
    'ShippedDate': '1998-01-13T00:00:00.000Z',
    'Freight': 306.07,
    'ShipName': 'Königlich Essen',
    'ShipAddress': 'Maubelstr. 90',
    'ShipCity': 'Brandenburg',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10818,
    'CustomerID': 'MAGAA',
    'OrderDate': '1998-01-07T00:00:00.000Z',
    'ShippedDate': '1998-01-12T00:00:00.000Z',
    'Freight': 65.48,
    'ShipName': 'Magazzini Alimentari Riuniti',
    'ShipAddress': 'Via Ludovico il Moro 22',
    'ShipCity': 'Bergamo',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10819,

```

```

        'CustomerID': 'CACTU',
        'OrderDate': '1998-01-07T00:00:00.000Z',
        'ShippedDate': '1998-01-16T00:00:00.000Z',
        'Freight': 19.76,
        'ShipName': 'Cactus Comidas para llevar',
        'ShipAddress': 'Cerrito 333',
        'ShipCity': 'Buenos Aires',
        'ShipRegion': null,
        'ShipCountry': 'Argentina'
    },
    {
        'OrderID': 10820,
        'CustomerID': 'RATTC',
        'OrderDate': '1998-01-07T00:00:00.000Z',
        'ShippedDate': '1998-01-13T00:00:00.000Z',
        'Freight': 37.52,
        'ShipName': 'Rattlesnake Canyon Grocery',
        'ShipAddress': '2817 Milton Dr.',
        'ShipCity': 'Albuquerque',
        'ShipRegion': 'NM',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10821,
        'CustomerID': 'SPLIR',
        'OrderDate': '1998-01-08T00:00:00.000Z',
        'ShippedDate': '1998-01-15T00:00:00.000Z',
        'Freight': 36.68,
        'ShipName': 'Split Rail Beer & Ale',
        'ShipAddress': 'P.O. Box 555',
        'ShipCity': 'Lander',
        'ShipRegion': 'WY',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10822,
        'CustomerID': 'TRAIH',
        'OrderDate': '1998-01-08T00:00:00.000Z',
        'ShippedDate': '1998-01-16T00:00:00.000Z',
        'Freight': 7,
        'ShipName': 'Trail\' Head Gourmet Provisioners',
        'ShipAddress': '722 DaVinci Blvd.',
        'ShipCity': 'Kirkland',
        'ShipRegion': 'WA',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10823,
        'CustomerID': 'LILAS',
        'OrderDate': '1998-01-09T00:00:00.000Z',
        'ShippedDate': '1998-01-13T00:00:00.000Z',
        'Freight': 163.97,
        'ShipName': 'LILA-Supermercado',
        'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
        'ShipCity': 'Barquisimeto',
        'ShipRegion': 'Lara',
        'ShipCountry': 'Venezuela'
    }

```

```

    },
    {
        'OrderID': 10824,
        'CustomerID': 'FOLKO',
        'OrderDate': '1998-01-09T00:00:00.000Z',
        'ShippedDate': '1998-01-30T00:00:00.000Z',
        'Freight': 1.23,
        'ShipName': 'Folk och fä HB',
        'ShipAddress': 'Åkergatan 24',
        'ShipCity': 'Bräcke',
        'ShipRegion': null,
        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 10825,
        'CustomerID': 'DRACD',
        'OrderDate': '1998-01-09T00:00:00.000Z',
        'ShippedDate': '1998-01-14T00:00:00.000Z',
        'Freight': 79.25,
        'ShipName': 'Drachenblut Delikatessen',
        'ShipAddress': 'Walserweg 21',
        'ShipCity': 'Aachen',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10826,
        'CustomerID': 'BLONP',
        'OrderDate': '1998-01-12T00:00:00.000Z',
        'ShippedDate': '1998-02-06T00:00:00.000Z',
        'Freight': 7.09,
        'ShipName': 'Blondel père et fils',
        'ShipAddress': '24, place Kléber',
        'ShipCity': 'Strasbourg',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10827,
        'CustomerID': 'BONAP',
        'OrderDate': '1998-01-12T00:00:00.000Z',
        'ShippedDate': '1998-02-06T00:00:00.000Z',
        'Freight': 63.54,
        'ShipName': 'Bon app',
        'ShipAddress': '12, rue des Bouchers',
        'ShipCity': 'Marseille',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10828,
        'CustomerID': 'RANCH',
        'OrderDate': '1998-01-13T00:00:00.000Z',
        'ShippedDate': '1998-02-04T00:00:00.000Z',
        'Freight': 90.85,
        'ShipName': 'Rancho grande',
        'ShipAddress': 'Av. del Libertador 900',
    }

```

```

        'ShipCity': 'Buenos Aires',
        'ShipRegion': null,
        'ShipCountry': 'Argentina'
    },
    {
        'OrderID': 10829,
        'CustomerID': 'ISLAT',
        'OrderDate': '1998-01-13T00:00:00.000Z',
        'ShippedDate': '1998-01-23T00:00:00.000Z',
        'Freight': 154.72,
        'ShipName': 'Island Trading',
        'ShipAddress': 'Garden House Crowther Way',
        'ShipCity': 'Cowes',
        'ShipRegion': 'Isle of Wight',
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10830,
        'CustomerID': 'TRADH',
        'OrderDate': '1998-01-13T00:00:00.000Z',
        'ShippedDate': '1998-01-21T00:00:00.000Z',
        'Freight': 81.83,
        'ShipName': 'Tradiçao Hipermercados',
        'ShipAddress': 'Av. Inês de Castro, 414',
        'ShipCity': 'Sao Paulo',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10831,
        'CustomerID': 'SANTG',
        'OrderDate': '1998-01-14T00:00:00.000Z',
        'ShippedDate': '1998-01-23T00:00:00.000Z',
        'Freight': 72.19,
        'ShipName': 'Santé Gourmet',
        'ShipAddress': 'Erling Skakkes gate 78',
        'ShipCity': 'Stavern',
        'ShipRegion': null,
        'ShipCountry': 'Norway'
    },
    {
        'OrderID': 10832,
        'CustomerID': 'LAMAI',
        'OrderDate': '1998-01-14T00:00:00.000Z',
        'ShippedDate': '1998-01-19T00:00:00.000Z',
        'Freight': 43.26,
        'ShipName': 'La maison d\'Asie',
        'ShipAddress': '1 rue Alsace-Lorraine',
        'ShipCity': 'Toulouse',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10833,
        'CustomerID': 'OTTIK',
        'OrderDate': '1998-01-15T00:00:00.000Z',
        'ShippedDate': '1998-01-23T00:00:00.000Z',

```



```

    'Freight': 71.49,
    'ShipName': 'Ottilies Käseladen',
    'ShipAddress': 'Mehrheimerstr. 369',
    'ShipCity': 'Köln',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10834,
    'CustomerID': 'TRADH',
    'OrderDate': '1998-01-15T00:00:00.000Z',
    'ShippedDate': '1998-01-19T00:00:00.000Z',
    'Freight': 29.78,
    'ShipName': 'Tradiçao Hipermercados',
    'ShipAddress': 'Av. Inês de Castro, 414',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10835,
    'CustomerID': 'ALFKI',
    'OrderDate': '1998-01-15T00:00:00.000Z',
    'ShippedDate': '1998-01-21T00:00:00.000Z',
    'Freight': 69.53,
    'ShipName': 'Alfred\' Futterkiste',
    'ShipAddress': 'Obere Str. 57',
    'ShipCity': 'Berlin',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10836,
    'CustomerID': 'ERNSH',
    'OrderDate': '1998-01-16T00:00:00.000Z',
    'ShippedDate': '1998-01-21T00:00:00.000Z',
    'Freight': 411.88,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10837,
    'CustomerID': 'BERGS',
    'OrderDate': '1998-01-16T00:00:00.000Z',
    'ShippedDate': '1998-01-23T00:00:00.000Z',
    'Freight': 13.32,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10838,

```

```

      'CustomerID': 'LINOD',
      'OrderDate': '1998-01-19T00:00:00.000Z',
      'ShippedDate': '1998-01-23T00:00:00.000Z',
      'Freight': 59.28,
      'ShipName': 'LINO-Delicateses',
      'ShipAddress': 'Ave. 5 de Mayo Porlamar',
      'ShipCity': 'I. de Margarita',
      'ShipRegion': 'Nueva Esparta',
      'ShipCountry': 'Venezuela'
    },
    {
      'OrderID': 10839,
      'CustomerID': 'TRADH',
      'OrderDate': '1998-01-19T00:00:00.000Z',
      'ShippedDate': '1998-01-22T00:00:00.000Z',
      'Freight': 35.43,
      'ShipName': 'Tradiçao Hipermercados',
      'ShipAddress': 'Av. Inês de Castro, 414',
      'ShipCity': 'Sao Paulo',
      'ShipRegion': 'SP',
      'ShipCountry': 'Brazil'
    },
    {
      'OrderID': 10840,
      'CustomerID': 'LINOD',
      'OrderDate': '1998-01-19T00:00:00.000Z',
      'ShippedDate': '1998-02-16T00:00:00.000Z',
      'Freight': 2.71,
      'ShipName': 'LINO-Delicateses',
      'ShipAddress': 'Ave. 5 de Mayo Porlamar',
      'ShipCity': 'I. de Margarita',
      'ShipRegion': 'Nueva Esparta',
      'ShipCountry': 'Venezuela'
    },
    {
      'OrderID': 10841,
      'CustomerID': 'SUPRD',
      'OrderDate': '1998-01-20T00:00:00.000Z',
      'ShippedDate': '1998-01-29T00:00:00.000Z',
      'Freight': 424.3,
      'ShipName': 'Suprêmes délices',
      'ShipAddress': 'Boulevard Tirou, 255',
      'ShipCity': 'Charleroi',
      'ShipRegion': null,
      'ShipCountry': 'Belgium'
    },
    {
      'OrderID': 10842,
      'CustomerID': 'TORTU',
      'OrderDate': '1998-01-20T00:00:00.000Z',
      'ShippedDate': '1998-01-29T00:00:00.000Z',
      'Freight': 54.42,
      'ShipName': 'Tortuga Restaurante',
      'ShipAddress': 'Avda. Azteca 123',
      'ShipCity': 'México D.F.',
      'ShipRegion': null,
      'ShipCountry': 'Mexico'
    }
  ]

```

```

},
{
  'OrderID': 10843,
  'CustomerID': 'VICTE',
  'OrderDate': '1998-01-21T00:00:00.000Z',
  'ShippedDate': '1998-01-26T00:00:00.000Z',
  'Freight': 9.26,
  'ShipName': 'Victuailles en stock',
  'ShipAddress': '2, rue du Commerce',
  'ShipCity': 'Lyon',
  'ShipRegion': null,
  'ShipCountry': 'France'
},
{
  'OrderID': 10844,
  'CustomerID': 'PICCO',
  'OrderDate': '1998-01-21T00:00:00.000Z',
  'ShippedDate': '1998-01-26T00:00:00.000Z',
  'Freight': 25.22,
  'ShipName': 'Piccolo und mehr',
  'ShipAddress': 'Geislweg 14',
  'ShipCity': 'Salzburg',
  'ShipRegion': null,
  'ShipCountry': 'Austria'
},
{
  'OrderID': 10845,
  'CustomerID': 'QUICK',
  'OrderDate': '1998-01-21T00:00:00.000Z',
  'ShippedDate': '1998-01-30T00:00:00.000Z',
  'Freight': 212.98,
  'ShipName': 'QUICK-Stop',
  'ShipAddress': 'Taucherstraße 10',
  'ShipCity': 'Cunewalde',
  'ShipRegion': null,
  'ShipCountry': 'Germany'
},
{
  'OrderID': 10846,
  'CustomerID': 'SUPRD',
  'OrderDate': '1998-01-22T00:00:00.000Z',
  'ShippedDate': '1998-01-23T00:00:00.000Z',
  'Freight': 56.46,
  'ShipName': 'Suprêmes délices',
  'ShipAddress': 'Boulevard Tirou, 255',
  'ShipCity': 'Charleroi',
  'ShipRegion': null,
  'ShipCountry': 'Belgium'
},
{
  'OrderID': 10847,
  'CustomerID': 'SAVEA',
  'OrderDate': '1998-01-22T00:00:00.000Z',
  'ShippedDate': '1998-02-10T00:00:00.000Z',
  'Freight': 487.57,
  'ShipName': 'Save-a-lot Markets',
  'ShipAddress': '187 Suffolk Ln.',

```

```

        'ShipCity': 'Boise',
        'ShipRegion': 'ID',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10848,
        'CustomerID': 'CONSH',
        'OrderDate': '1998-01-23T00:00:00.000Z',
        'ShippedDate': '1998-01-29T00:00:00.000Z',
        'Freight': 38.24,
        'ShipName': 'Consolidated Holdings',
        'ShipAddress': 'Berkeley Gardens 12 Brewery',
        'ShipCity': 'London',
        'ShipRegion': null,
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10849,
        'CustomerID': 'KOENE',
        'OrderDate': '1998-01-23T00:00:00.000Z',
        'ShippedDate': '1998-01-30T00:00:00.000Z',
        'Freight': 0.56,
        'ShipName': 'Königlich Essen',
        'ShipAddress': 'Maubelstr. 90',
        'ShipCity': 'Brandenburg',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10850,
        'CustomerID': 'VICTE',
        'OrderDate': '1998-01-23T00:00:00.000Z',
        'ShippedDate': '1998-01-30T00:00:00.000Z',
        'Freight': 49.19,
        'ShipName': 'Victuailles en stock',
        'ShipAddress': '2, rue du Commerce',
        'ShipCity': 'Lyon',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10851,
        'CustomerID': 'RICAR',
        'OrderDate': '1998-01-26T00:00:00.000Z',
        'ShippedDate': '1998-02-02T00:00:00.000Z',
        'Freight': 160.55,
        'ShipName': 'Ricardo Adocicados',
        'ShipAddress': 'Av. Copacabana, 267',
        'ShipCity': 'Rio de Janeiro',
        'ShipRegion': 'RJ',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10852,
        'CustomerID': 'RATTC',
        'OrderDate': '1998-01-26T00:00:00.000Z',
        'ShippedDate': '1998-01-30T00:00:00.000Z',

```

```

    'Freight': 174.05,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10853,
    'CustomerID': 'BLAUS',
    'OrderDate': '1998-01-27T00:00:00.000Z',
    'ShippedDate': '1998-02-03T00:00:00.000Z',
    'Freight': 53.83,
    'ShipName': 'Blauer See Delikatessen',
    'ShipAddress': 'Forsterstr. 57',
    'ShipCity': 'Mannheim',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10854,
    'CustomerID': 'ERNSH',
    'OrderDate': '1998-01-27T00:00:00.000Z',
    'ShippedDate': '1998-02-05T00:00:00.000Z',
    'Freight': 100.22,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10855,
    'CustomerID': 'OLDWO',
    'OrderDate': '1998-01-27T00:00:00.000Z',
    'ShippedDate': '1998-02-04T00:00:00.000Z',
    'Freight': 170.97,
    'ShipName': 'Old World Delicatessen',
    'ShipAddress': '2743 Bering St.',
    'ShipCity': 'Anchorage',
    'ShipRegion': 'AK',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10856,
    'CustomerID': 'ANTON',
    'OrderDate': '1998-01-28T00:00:00.000Z',
    'ShippedDate': '1998-02-10T00:00:00.000Z',
    'Freight': 58.43,
    'ShipName': 'Antonio Moreno Taquería',
    'ShipAddress': 'Mataderos 2312',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10857,

```

```

    'CustomerID': 'BERGS',
    'OrderDate': '1998-01-28T00:00:00.000Z',
    'ShippedDate': '1998-02-06T00:00:00.000Z',
    'Freight': 188.85,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10858,
    'CustomerID': 'LACOR',
    'OrderDate': '1998-01-29T00:00:00.000Z',
    'ShippedDate': '1998-02-03T00:00:00.000Z',
    'Freight': 52.51,
    'ShipName': 'La corne d\'abondance',
    'ShipAddress': '67, avenue de l\'Europe',
    'ShipCity': 'Versailles',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10859,
    'CustomerID': 'FRANK',
    'OrderDate': '1998-01-29T00:00:00.000Z',
    'ShippedDate': '1998-02-02T00:00:00.000Z',
    'Freight': 76.1,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10860,
    'CustomerID': 'FRANR',
    'OrderDate': '1998-01-29T00:00:00.000Z',
    'ShippedDate': '1998-02-04T00:00:00.000Z',
    'Freight': 19.26,
    'ShipName': 'France restauration',
    'ShipAddress': '54, rue Royale',
    'ShipCity': 'Nantes',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10861,
    'CustomerID': 'WHITC',
    'OrderDate': '1998-01-30T00:00:00.000Z',
    'ShippedDate': '1998-02-17T00:00:00.000Z',
    'Freight': 14.93,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  }

```

```

    },
    {
        'OrderID': 10862,
        'CustomerID': 'LEHMS',
        'OrderDate': '1998-01-30T00:00:00.000Z',
        'ShippedDate': '1998-02-02T00:00:00.000Z',
        'Freight': 53.23,
        'ShipName': 'Lehmanns Marktstand',
        'ShipAddress': 'Magazinweg 7',
        'ShipCity': 'Frankfurt a.M.',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10863,
        'CustomerID': 'HILAA',
        'OrderDate': '1998-02-02T00:00:00.000Z',
        'ShippedDate': '1998-02-17T00:00:00.000Z',
        'Freight': 30.26,
        'ShipName': 'HILARION-Abastos',
        'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
        'ShipCity': 'San Cristóbal',
        'ShipRegion': 'Táchira',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10864,
        'CustomerID': 'AROUT',
        'OrderDate': '1998-02-02T00:00:00.000Z',
        'ShippedDate': '1998-02-09T00:00:00.000Z',
        'Freight': 3.04,
        'ShipName': 'Around the Horn',
        'ShipAddress': 'Brook Farm Stratford St. Mary',
        'ShipCity': 'Colchester',
        'ShipRegion': 'Essex',
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10865,
        'CustomerID': 'QUICK',
        'OrderDate': '1998-02-02T00:00:00.000Z',
        'ShippedDate': '1998-02-12T00:00:00.000Z',
        'Freight': 348.14,
        'ShipName': 'QUICK-Stop',
        'ShipAddress': 'Taucherstraße 10',
        'ShipCity': 'Cunewalde',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10866,
        'CustomerID': 'BERGS',
        'OrderDate': '1998-02-03T00:00:00.000Z',
        'ShippedDate': '1998-02-12T00:00:00.000Z',
        'Freight': 109.11,
        'ShipName': 'Berglunds snabbköp',
        'ShipAddress': 'Berguvsvägen 8',
    }

```

```

    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10867,
    'CustomerID': 'LONEP',
    'OrderDate': '1998-02-03T00:00:00.000Z',
    'ShippedDate': '1998-02-11T00:00:00.000Z',
    'Freight': 1.93,
    'ShipName': 'Lonesome Pine Restaurant',
    'ShipAddress': '89 Chiaroscuro Rd.',
    'ShipCity': 'Portland',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10868,
    'CustomerID': 'QUEEN',
    'OrderDate': '1998-02-04T00:00:00.000Z',
    'ShippedDate': '1998-02-23T00:00:00.000Z',
    'Freight': 191.27,
    'ShipName': 'Queen Cozinha',
    'ShipAddress': 'Alameda dos Canários, 891',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10869,
    'CustomerID': 'SEVES',
    'OrderDate': '1998-02-04T00:00:00.000Z',
    'ShippedDate': '1998-02-09T00:00:00.000Z',
    'Freight': 143.28,
    'ShipName': 'Seven Seas Imports',
    'ShipAddress': '90 Wadhurst Rd.',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10870,
    'CustomerID': 'WOLZA',
    'OrderDate': '1998-02-04T00:00:00.000Z',
    'ShippedDate': '1998-02-13T00:00:00.000Z',
    'Freight': 12.04,
    'ShipName': 'Wolski Zajazd',
    'ShipAddress': 'ul. Filtrowa 68',
    'ShipCity': 'Warszawa',
    'ShipRegion': null,
    'ShipCountry': 'Poland'
  },
  {
    'OrderID': 10871,
    'CustomerID': 'BONAP',
    'OrderDate': '1998-02-05T00:00:00.000Z',
    'ShippedDate': '1998-02-10T00:00:00.000Z',

```



```

    'Freight': 112.27,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10872,
    'CustomerID': 'GODOS',
    'OrderDate': '1998-02-05T00:00:00.000Z',
    'ShippedDate': '1998-02-09T00:00:00.000Z',
    'Freight': 175.32,
    'ShipName': 'Godos Cocina Típica',
    'ShipAddress': 'C/ Romero, 33',
    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10873,
    'CustomerID': 'WILMK',
    'OrderDate': '1998-02-06T00:00:00.000Z',
    'ShippedDate': '1998-02-09T00:00:00.000Z',
    'Freight': 0.82,
    'ShipName': 'Wilman Kala',
    'ShipAddress': 'Keskuskatu 45',
    'ShipCity': 'Helsinki',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10874,
    'CustomerID': 'GODOS',
    'OrderDate': '1998-02-06T00:00:00.000Z',
    'ShippedDate': '1998-02-11T00:00:00.000Z',
    'Freight': 19.58,
    'ShipName': 'Godos Cocina Típica',
    'ShipAddress': 'C/ Romero, 33',
    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10875,
    'CustomerID': 'BERGS',
    'OrderDate': '1998-02-06T00:00:00.000Z',
    'ShippedDate': '1998-03-03T00:00:00.000Z',
    'Freight': 32.37,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10876,

```

```

    'CustomerID': 'BONAP',
    'OrderDate': '1998-02-09T00:00:00.000Z',
    'ShippedDate': '1998-02-12T00:00:00.000Z',
    'Freight': 60.42,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10877,
    'CustomerID': 'RICAR',
    'OrderDate': '1998-02-09T00:00:00.000Z',
    'ShippedDate': '1998-02-19T00:00:00.000Z',
    'Freight': 38.06,
    'ShipName': 'Ricardo Adocicados',
    'ShipAddress': 'Av. Copacabana, 267',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10878,
    'CustomerID': 'QUICK',
    'OrderDate': '1998-02-10T00:00:00.000Z',
    'ShippedDate': '1998-02-12T00:00:00.000Z',
    'Freight': 46.69,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10879,
    'CustomerID': 'WILMK',
    'OrderDate': '1998-02-10T00:00:00.000Z',
    'ShippedDate': '1998-02-12T00:00:00.000Z',
    'Freight': 8.5,
    'ShipName': 'Wilman Kala',
    'ShipAddress': 'Keskuskatu 45',
    'ShipCity': 'Helsinki',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10880,
    'CustomerID': 'FOLKO',
    'OrderDate': '1998-02-10T00:00:00.000Z',
    'ShippedDate': '1998-02-18T00:00:00.000Z',
    'Freight': 88.01,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  }

```

```

},
{
  'OrderID': 10881,
  'CustomerID': 'CACTU',
  'OrderDate': '1998-02-11T00:00:00.000Z',
  'ShippedDate': '1998-02-18T00:00:00.000Z',
  'Freight': 2.84,
  'ShipName': 'Cactus Comidas para llevar',
  'ShipAddress': 'Cerrito 333',
  'ShipCity': 'Buenos Aires',
  'ShipRegion': null,
  'ShipCountry': 'Argentina'
},
{
  'OrderID': 10882,
  'CustomerID': 'SAVEA',
  'OrderDate': '1998-02-11T00:00:00.000Z',
  'ShippedDate': '1998-02-20T00:00:00.000Z',
  'Freight': 23.1,
  'ShipName': 'Save-a-lot Markets',
  'ShipAddress': '187 Suffolk Ln.',
  'ShipCity': 'Boise',
  'ShipRegion': 'ID',
  'ShipCountry': 'USA'
},
{
  'OrderID': 10883,
  'CustomerID': 'LONEP',
  'OrderDate': '1998-02-12T00:00:00.000Z',
  'ShippedDate': '1998-02-20T00:00:00.000Z',
  'Freight': 0.53,
  'ShipName': 'Lonesome Pine Restaurant',
  'ShipAddress': '89 Chiaroscuro Rd.',
  'ShipCity': 'Portland',
  'ShipRegion': 'OR',
  'ShipCountry': 'USA'
},
{
  'OrderID': 10884,
  'CustomerID': 'LETSS',
  'OrderDate': '1998-02-12T00:00:00.000Z',
  'ShippedDate': '1998-02-13T00:00:00.000Z',
  'Freight': 90.97,
  'ShipName': 'Let\ Stop N Shop',
  'ShipAddress': '87 Polk St. Suite 5',
  'ShipCity': 'San Francisco',
  'ShipRegion': 'CA',
  'ShipCountry': 'USA'
},
{
  'OrderID': 10885,
  'CustomerID': 'SUPRD',
  'OrderDate': '1998-02-12T00:00:00.000Z',
  'ShippedDate': '1998-02-18T00:00:00.000Z',
  'Freight': 5.64,
  'ShipName': 'Suprêmes délices',
  'ShipAddress': 'Boulevard Tirou, 255',

```

```

    'ShipCity': 'Charleroi',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10886,
    'CustomerID': 'HANAR',
    'OrderDate': '1998-02-13T00:00:00.000Z',
    'ShippedDate': '1998-03-02T00:00:00.000Z',
    'Freight': 4.99,
    'ShipName': 'Hanari Carnes',
    'ShipAddress': 'Rua do Paço, 67',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10887,
    'CustomerID': 'GALED',
    'OrderDate': '1998-02-13T00:00:00.000Z',
    'ShippedDate': '1998-02-16T00:00:00.000Z',
    'Freight': 1.25,
    'ShipName': 'Galería del gastronómo',
    'ShipAddress': 'Rambla de Cataluña, 23',
    'ShipCity': 'Barcelona',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10888,
    'CustomerID': 'GODOS',
    'OrderDate': '1998-02-16T00:00:00.000Z',
    'ShippedDate': '1998-02-23T00:00:00.000Z',
    'Freight': 51.87,
    'ShipName': 'Godos Cocina Típica',
    'ShipAddress': 'C/ Romero, 33',
    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10889,
    'CustomerID': 'RATTC',
    'OrderDate': '1998-02-16T00:00:00.000Z',
    'ShippedDate': '1998-02-23T00:00:00.000Z',
    'Freight': 280.61,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10890,
    'CustomerID': 'DUMON',
    'OrderDate': '1998-02-16T00:00:00.000Z',
    'ShippedDate': '1998-02-18T00:00:00.000Z',

```

```

    'Freight': 32.76,
    'ShipName': 'Du monde entier',
    'ShipAddress': '67, rue des Cinquante Otages',
    'ShipCity': 'Nantes',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10891,
    'CustomerID': 'LEHMS',
    'OrderDate': '1998-02-17T00:00:00.000Z',
    'ShippedDate': '1998-02-19T00:00:00.000Z',
    'Freight': 20.37,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10892,
    'CustomerID': 'MAISD',
    'OrderDate': '1998-02-17T00:00:00.000Z',
    'ShippedDate': '1998-02-19T00:00:00.000Z',
    'Freight': 120.27,
    'ShipName': 'Maison Dewey',
    'ShipAddress': 'Rue Joseph-Bens 532',
    'ShipCity': 'Bruxelles',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10893,
    'CustomerID': 'KOENE',
    'OrderDate': '1998-02-18T00:00:00.000Z',
    'ShippedDate': '1998-02-20T00:00:00.000Z',
    'Freight': 77.78,
    'ShipName': 'Königlich Essen',
    'ShipAddress': 'Maubelstr. 90',
    'ShipCity': 'Brandenburg',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10894,
    'CustomerID': 'SAVEA',
    'OrderDate': '1998-02-18T00:00:00.000Z',
    'ShippedDate': '1998-02-20T00:00:00.000Z',
    'Freight': 116.13,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10895,

```

```

    'CustomerID': 'ERNSH',
    'OrderDate': '1998-02-18T00:00:00.000Z',
    'ShippedDate': '1998-02-23T00:00:00.000Z',
    'Freight': 162.75,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10896,
    'CustomerID': 'MAISD',
    'OrderDate': '1998-02-19T00:00:00.000Z',
    'ShippedDate': '1998-02-27T00:00:00.000Z',
    'Freight': 32.45,
    'ShipName': 'Maison Dewey',
    'ShipAddress': 'Rue Joseph-Bens 532',
    'ShipCity': 'Bruxelles',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10897,
    'CustomerID': 'HUNGO',
    'OrderDate': '1998-02-19T00:00:00.000Z',
    'ShippedDate': '1998-02-25T00:00:00.000Z',
    'Freight': 603.54,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10898,
    'CustomerID': 'OCEAN',
    'OrderDate': '1998-02-20T00:00:00.000Z',
    'ShippedDate': '1998-03-06T00:00:00.000Z',
    'Freight': 1.27,
    'ShipName': 'Océano Atlántico Ltda.',
    'ShipAddress': 'Ing. Gustavo Moncada 8585 Piso 20-A',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 10899,
    'CustomerID': 'LILAS',
    'OrderDate': '1998-02-20T00:00:00.000Z',
    'ShippedDate': '1998-02-26T00:00:00.000Z',
    'Freight': 1.21,
    'ShipName': 'LILA-Supermercado',
    'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  }

```

```

},
{
  'OrderID': 10900,
  'CustomerID': 'WELLI',
  'OrderDate': '1998-02-20T00:00:00.000Z',
  'ShippedDate': '1998-03-04T00:00:00.000Z',
  'Freight': 1.66,
  'ShipName': 'Wellington Importadora',
  'ShipAddress': 'Rua do Mercado, 12',
  'ShipCity': 'Resende',
  'ShipRegion': 'SP',
  'ShipCountry': 'Brazil'
},
{
  'OrderID': 10901,
  'CustomerID': 'HILAA',
  'OrderDate': '1998-02-23T00:00:00.000Z',
  'ShippedDate': '1998-02-26T00:00:00.000Z',
  'Freight': 62.09,
  'ShipName': 'HILARION-Abastos',
  'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
  'ShipCity': 'San Cristóbal',
  'ShipRegion': 'Táchira',
  'ShipCountry': 'Venezuela'
},
{
  'OrderID': 10902,
  'CustomerID': 'FOLKO',
  'OrderDate': '1998-02-23T00:00:00.000Z',
  'ShippedDate': '1998-03-03T00:00:00.000Z',
  'Freight': 44.15,
  'ShipName': 'Folk och fä HB',
  'ShipAddress': 'Åkergatan 24',
  'ShipCity': 'Bräcke',
  'ShipRegion': null,
  'ShipCountry': 'Sweden'
},
{
  'OrderID': 10903,
  'CustomerID': 'HANAR',
  'OrderDate': '1998-02-24T00:00:00.000Z',
  'ShippedDate': '1998-03-04T00:00:00.000Z',
  'Freight': 36.71,
  'ShipName': 'Hanari Carnes',
  'ShipAddress': 'Rua do Paço, 67',
  'ShipCity': 'Rio de Janeiro',
  'ShipRegion': 'RJ',
  'ShipCountry': 'Brazil'
},
{
  'OrderID': 10904,
  'CustomerID': 'WHITC',
  'OrderDate': '1998-02-24T00:00:00.000Z',
  'ShippedDate': '1998-02-27T00:00:00.000Z',
  'Freight': 162.95,
  'ShipName': 'White Clover Markets',
  'ShipAddress': '1029 - 12th Ave. S.',

```

```

    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10905,
    'CustomerID': 'WELLI',
    'OrderDate': '1998-02-24T00:00:00.000Z',
    'ShippedDate': '1998-03-06T00:00:00.000Z',
    'Freight': 13.72,
    'ShipName': 'Wellington Importadora',
    'ShipAddress': 'Rua do Mercado, 12',
    'ShipCity': 'Resende',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10906,
    'CustomerID': 'WOLZA',
    'OrderDate': '1998-02-25T00:00:00.000Z',
    'ShippedDate': '1998-03-03T00:00:00.000Z',
    'Freight': 26.29,
    'ShipName': 'Wolski Zajazd',
    'ShipAddress': 'ul. Filtrowa 68',
    'ShipCity': 'Warszawa',
    'ShipRegion': null,
    'ShipCountry': 'Poland'
  },
  {
    'OrderID': 10907,
    'CustomerID': 'SPECB',
    'OrderDate': '1998-02-25T00:00:00.000Z',
    'ShippedDate': '1998-02-27T00:00:00.000Z',
    'Freight': 9.19,
    'ShipName': 'Spécialités du monde',
    'ShipAddress': '25, rue Lauriston',
    'ShipCity': 'Paris',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10908,
    'CustomerID': 'REGGC',
    'OrderDate': '1998-02-26T00:00:00.000Z',
    'ShippedDate': '1998-03-06T00:00:00.000Z',
    'Freight': 32.96,
    'ShipName': 'Reggiani Caseifici',
    'ShipAddress': 'Strada Provinciale 124',
    'ShipCity': 'Reggio Emilia',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10909,
    'CustomerID': 'SANTG',
    'OrderDate': '1998-02-26T00:00:00.000Z',
    'ShippedDate': '1998-03-10T00:00:00.000Z',

```



```

    'Freight': 53.05,
    'ShipName': 'Santé Gourmet',
    'ShipAddress': 'Erling Skakkess gate 78',
    'ShipCity': 'Stavern',
    'ShipRegion': null,
    'ShipCountry': 'Norway'
  },
  {
    'OrderID': 10910,
    'CustomerID': 'WILMK',
    'OrderDate': '1998-02-26T00:00:00.000Z',
    'ShippedDate': '1998-03-04T00:00:00.000Z',
    'Freight': 38.11,
    'ShipName': 'Wilman Kala',
    'ShipAddress': 'Keskuskatu 45',
    'ShipCity': 'Helsinki',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10911,
    'CustomerID': 'GODOS',
    'OrderDate': '1998-02-26T00:00:00.000Z',
    'ShippedDate': '1998-03-05T00:00:00.000Z',
    'Freight': 38.19,
    'ShipName': 'Godos Cocina Típica',
    'ShipAddress': 'C/ Romero, 33',
    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10912,
    'CustomerID': 'HUNGO',
    'OrderDate': '1998-02-26T00:00:00.000Z',
    'ShippedDate': '1998-03-18T00:00:00.000Z',
    'Freight': 580.91,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10913,
    'CustomerID': 'QUEEN',
    'OrderDate': '1998-02-26T00:00:00.000Z',
    'ShippedDate': '1998-03-04T00:00:00.000Z',
    'Freight': 33.05,
    'ShipName': 'Queen Cozinha',
    'ShipAddress': 'Alameda dos Canários, 891',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10914,

```

```

      'CustomerID': 'QUEEN',
      'OrderDate': '1998-02-27T00:00:00.000Z',
      'ShippedDate': '1998-03-02T00:00:00.000Z',
      'Freight': 21.19,
      'ShipName': 'Queen Cozinha',
      'ShipAddress': 'Alameda dos Canários, 891',
      'ShipCity': 'Sao Paulo',
      'ShipRegion': 'SP',
      'ShipCountry': 'Brazil'
    },
    {
      'OrderID': 10915,
      'CustomerID': 'TORTU',
      'OrderDate': '1998-02-27T00:00:00.000Z',
      'ShippedDate': '1998-03-02T00:00:00.000Z',
      'Freight': 3.51,
      'ShipName': 'Tortuga Restaurante',
      'ShipAddress': 'Avda. Azteca 123',
      'ShipCity': 'México D.F.',
      'ShipRegion': null,
      'ShipCountry': 'Mexico'
    },
    {
      'OrderID': 10916,
      'CustomerID': 'RANCH',
      'OrderDate': '1998-02-27T00:00:00.000Z',
      'ShippedDate': '1998-03-09T00:00:00.000Z',
      'Freight': 63.77,
      'ShipName': 'Rancho grande',
      'ShipAddress': 'Av. del Libertador 900',
      'ShipCity': 'Buenos Aires',
      'ShipRegion': null,
      'ShipCountry': 'Argentina'
    },
    {
      'OrderID': 10917,
      'CustomerID': 'ROMEY',
      'OrderDate': '1998-03-02T00:00:00.000Z',
      'ShippedDate': '1998-03-11T00:00:00.000Z',
      'Freight': 8.29,
      'ShipName': 'Romero y tomillo',
      'ShipAddress': 'Gran Vía, 1',
      'ShipCity': 'Madrid',
      'ShipRegion': null,
      'ShipCountry': 'Spain'
    },
    {
      'OrderID': 10918,
      'CustomerID': 'BOTTM',
      'OrderDate': '1998-03-02T00:00:00.000Z',
      'ShippedDate': '1998-03-11T00:00:00.000Z',
      'Freight': 48.83,
      'ShipName': 'Bottom-Dollar Markets',
      'ShipAddress': '23 Tsawassen Blvd.',
      'ShipCity': 'Tsawassen',
      'ShipRegion': 'BC',
      'ShipCountry': 'Canada'
    }
  ]

```

```

    },
    {
        'OrderID': 10919,
        'CustomerID': 'LINOD',
        'OrderDate': '1998-03-02T00:00:00.000Z',
        'ShippedDate': '1998-03-04T00:00:00.000Z',
        'Freight': 19.8,
        'ShipName': 'LINO-Delicateses',
        'ShipAddress': 'Ave. 5 de Mayo Porlamar',
        'ShipCity': 'I. de Margarita',
        'ShipRegion': 'Nueva Esparta',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10920,
        'CustomerID': 'AROUT',
        'OrderDate': '1998-03-03T00:00:00.000Z',
        'ShippedDate': '1998-03-09T00:00:00.000Z',
        'Freight': 29.61,
        'ShipName': 'Around the Horn',
        'ShipAddress': 'Brook Farm Stratford St. Mary',
        'ShipCity': 'Colchester',
        'ShipRegion': 'Essex',
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10921,
        'CustomerID': 'VAFFE',
        'OrderDate': '1998-03-03T00:00:00.000Z',
        'ShippedDate': '1998-03-09T00:00:00.000Z',
        'Freight': 176.48,
        'ShipName': 'Vaffeljernet',
        'ShipAddress': 'Smagsloget 45',
        'ShipCity': 'Århus',
        'ShipRegion': null,
        'ShipCountry': 'Denmark'
    },
    {
        'OrderID': 10922,
        'CustomerID': 'HANAR',
        'OrderDate': '1998-03-03T00:00:00.000Z',
        'ShippedDate': '1998-03-05T00:00:00.000Z',
        'Freight': 62.74,
        'ShipName': 'Hanari Carnes',
        'ShipAddress': 'Rua do Paço, 67',
        'ShipCity': 'Rio de Janeiro',
        'ShipRegion': 'RJ',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10923,
        'CustomerID': 'LAMAI',
        'OrderDate': '1998-03-03T00:00:00.000Z',
        'ShippedDate': '1998-03-13T00:00:00.000Z',
        'Freight': 68.26,
        'ShipName': 'La maison d\'Asie',
        'ShipAddress': '1 rue Alsace-Lorraine',
    }

```

```

    'ShipCity': 'Toulouse',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10924,
    'CustomerID': 'BERGS',
    'OrderDate': '1998-03-04T00:00:00.000Z',
    'ShippedDate': '1998-04-08T00:00:00.000Z',
    'Freight': 151.52,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10925,
    'CustomerID': 'HANAR',
    'OrderDate': '1998-03-04T00:00:00.000Z',
    'ShippedDate': '1998-03-13T00:00:00.000Z',
    'Freight': 2.27,
    'ShipName': 'Hanari Carnes',
    'ShipAddress': 'Rua do Paço, 67',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10926,
    'CustomerID': 'ANATR',
    'OrderDate': '1998-03-04T00:00:00.000Z',
    'ShippedDate': '1998-03-11T00:00:00.000Z',
    'Freight': 39.92,
    'ShipName': 'Ana Trujillo Emparedados y helados',
    'ShipAddress': 'Avda. de la Constitución 2222',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10927,
    'CustomerID': 'LACOR',
    'OrderDate': '1998-03-05T00:00:00.000Z',
    'ShippedDate': '1998-04-08T00:00:00.000Z',
    'Freight': 19.79,
    'ShipName': 'La corne d\'abondance',
    'ShipAddress': '67, avenue de l\'Europe',
    'ShipCity': 'Versailles',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10928,
    'CustomerID': 'GALED',
    'OrderDate': '1998-03-05T00:00:00.000Z',
    'ShippedDate': '1998-03-18T00:00:00.000Z',

```

```

    'Freight': 1.36,
    'ShipName': 'Galería del gastronómo',
    'ShipAddress': 'Rambla de Cataluña, 23',
    'ShipCity': 'Barcelona',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10929,
    'CustomerID': 'FRANK',
    'OrderDate': '1998-03-05T00:00:00.000Z',
    'ShippedDate': '1998-03-12T00:00:00.000Z',
    'Freight': 33.93,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10930,
    'CustomerID': 'SUPRD',
    'OrderDate': '1998-03-06T00:00:00.000Z',
    'ShippedDate': '1998-03-18T00:00:00.000Z',
    'Freight': 15.55,
    'ShipName': 'Suprêmes délices',
    'ShipAddress': 'Boulevard Tirou, 255',
    'ShipCity': 'Charleroi',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10931,
    'CustomerID': 'RICSU',
    'OrderDate': '1998-03-06T00:00:00.000Z',
    'ShippedDate': '1998-03-19T00:00:00.000Z',
    'Freight': 13.6,
    'ShipName': 'Richter Supermarkt',
    'ShipAddress': 'Starenweg 5',
    'ShipCity': 'Genève',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10932,
    'CustomerID': 'BONAP',
    'OrderDate': '1998-03-06T00:00:00.000Z',
    'ShippedDate': '1998-03-24T00:00:00.000Z',
    'Freight': 134.64,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10933,

```

```

    'CustomerID': 'ISLAT',
    'OrderDate': '1998-03-06T00:00:00.000Z',
    'ShippedDate': '1998-03-16T00:00:00.000Z',
    'Freight': 54.15,
    'ShipName': 'Island Trading',
    'ShipAddress': 'Garden House Crowther Way',
    'ShipCity': 'Cowes',
    'ShipRegion': 'Isle of Wight',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10934,
    'CustomerID': 'LEHMS',
    'OrderDate': '1998-03-09T00:00:00.000Z',
    'ShippedDate': '1998-03-12T00:00:00.000Z',
    'Freight': 32.01,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10935,
    'CustomerID': 'WELLI',
    'OrderDate': '1998-03-09T00:00:00.000Z',
    'ShippedDate': '1998-03-18T00:00:00.000Z',
    'Freight': 47.59,
    'ShipName': 'Wellington Importadora',
    'ShipAddress': 'Rua do Mercado, 12',
    'ShipCity': 'Resende',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10936,
    'CustomerID': 'GREAL',
    'OrderDate': '1998-03-09T00:00:00.000Z',
    'ShippedDate': '1998-03-18T00:00:00.000Z',
    'Freight': 33.68,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10937,
    'CustomerID': 'CACTU',
    'OrderDate': '1998-03-10T00:00:00.000Z',
    'ShippedDate': '1998-03-13T00:00:00.000Z',
    'Freight': 31.51,
    'ShipName': 'Cactus Comidas para llevar',
    'ShipAddress': 'Cerrito 333',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  }

```

```

},
{
  'OrderID': 10938,
  'CustomerID': 'QUICK',
  'OrderDate': '1998-03-10T00:00:00.000Z',
  'ShippedDate': '1998-03-16T00:00:00.000Z',
  'Freight': 31.89,
  'ShipName': 'QUICK-Stop',
  'ShipAddress': 'Taucherstraße 10',
  'ShipCity': 'Cunewalde',
  'ShipRegion': null,
  'ShipCountry': 'Germany'
},
{
  'OrderID': 10939,
  'CustomerID': 'MAGAA',
  'OrderDate': '1998-03-10T00:00:00.000Z',
  'ShippedDate': '1998-03-13T00:00:00.000Z',
  'Freight': 76.33,
  'ShipName': 'Magazzini Alimentari Riuniti',
  'ShipAddress': 'Via Ludovico il Moro 22',
  'ShipCity': 'Bergamo',
  'ShipRegion': null,
  'ShipCountry': 'Italy'
},
{
  'OrderID': 10940,
  'CustomerID': 'BONAP',
  'OrderDate': '1998-03-11T00:00:00.000Z',
  'ShippedDate': '1998-03-23T00:00:00.000Z',
  'Freight': 19.77,
  'ShipName': 'Bon app',
  'ShipAddress': '12, rue des Bouchers',
  'ShipCity': 'Marseille',
  'ShipRegion': null,
  'ShipCountry': 'France'
},
{
  'OrderID': 10941,
  'CustomerID': 'SAVEA',
  'OrderDate': '1998-03-11T00:00:00.000Z',
  'ShippedDate': '1998-03-20T00:00:00.000Z',
  'Freight': 400.81,
  'ShipName': 'Save-a-lot Markets',
  'ShipAddress': '187 Suffolk Ln.',
  'ShipCity': 'Boise',
  'ShipRegion': 'ID',
  'ShipCountry': 'USA'
},
{
  'OrderID': 10942,
  'CustomerID': 'REGGC',
  'OrderDate': '1998-03-11T00:00:00.000Z',
  'ShippedDate': '1998-03-18T00:00:00.000Z',
  'Freight': 17.95,
  'ShipName': 'Reggiani Caseifici',
  'ShipAddress': 'Strada Provinciale 124',

```

```

    'ShipCity': 'Reggio Emilia',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10943,
    'CustomerID': 'BSBEV',
    'OrderDate': '1998-03-11T00:00:00.000Z',
    'ShippedDate': '1998-03-19T00:00:00.000Z',
    'Freight': 2.17,
    'ShipName': 'B\ Beverages',
    'ShipAddress': 'Fauntleroy Circus',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10944,
    'CustomerID': 'BOTTM',
    'OrderDate': '1998-03-12T00:00:00.000Z',
    'ShippedDate': '1998-03-13T00:00:00.000Z',
    'Freight': 52.92,
    'ShipName': 'Bottom-Dollar Markets',
    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10945,
    'CustomerID': 'MORGK',
    'OrderDate': '1998-03-12T00:00:00.000Z',
    'ShippedDate': '1998-03-18T00:00:00.000Z',
    'Freight': 10.22,
    'ShipName': 'Morgenstern Gesundkost',
    'ShipAddress': 'Heerstr. 22',
    'ShipCity': 'Leipzig',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10946,
    'CustomerID': 'VAFFE',
    'OrderDate': '1998-03-12T00:00:00.000Z',
    'ShippedDate': '1998-03-19T00:00:00.000Z',
    'Freight': 27.2,
    'ShipName': 'Vaffeljernet',
    'ShipAddress': 'Smagsloget 45',
    'ShipCity': 'Århus',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10947,
    'CustomerID': 'BSBEV',
    'OrderDate': '1998-03-13T00:00:00.000Z',
    'ShippedDate': '1998-03-16T00:00:00.000Z',

```



```

    'Freight': 3.26,
    'ShipName': 'B\' Beverages',
    'ShipAddress': 'Fauntleroy Circus',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10948,
    'CustomerID': 'GODOS',
    'OrderDate': '1998-03-13T00:00:00.000Z',
    'ShippedDate': '1998-03-19T00:00:00.000Z',
    'Freight': 23.39,
    'ShipName': 'Godos Cocina Típica',
    'ShipAddress': 'C/ Romero, 33',
    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10949,
    'CustomerID': 'BOTTM',
    'OrderDate': '1998-03-13T00:00:00.000Z',
    'ShippedDate': '1998-03-17T00:00:00.000Z',
    'Freight': 74.44,
    'ShipName': 'Bottom-Dollar Markets',
    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10950,
    'CustomerID': 'MAGAA',
    'OrderDate': '1998-03-16T00:00:00.000Z',
    'ShippedDate': '1998-03-23T00:00:00.000Z',
    'Freight': 2.5,
    'ShipName': 'Magazzini Alimentari Riuniti',
    'ShipAddress': 'Via Ludovico il Moro 22',
    'ShipCity': 'Bergamo',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10951,
    'CustomerID': 'RICSU',
    'OrderDate': '1998-03-16T00:00:00.000Z',
    'ShippedDate': '1998-04-07T00:00:00.000Z',
    'Freight': 30.85,
    'ShipName': 'Richter Supermarkt',
    'ShipAddress': 'Starenweg 5',
    'ShipCity': 'Genève',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10952,

```

```

    'CustomerID': 'ALFKI',
    'OrderDate': '1998-03-16T00:00:00.000Z',
    'ShippedDate': '1998-03-24T00:00:00.000Z',
    'Freight': 40.42,
    'ShipName': 'Alfred\ Futterkiste',
    'ShipAddress': 'Obere Str. 57',
    'ShipCity': 'Berlin',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10953,
    'CustomerID': 'AROUT',
    'OrderDate': '1998-03-16T00:00:00.000Z',
    'ShippedDate': '1998-03-25T00:00:00.000Z',
    'Freight': 23.72,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10954,
    'CustomerID': 'LINOD',
    'OrderDate': '1998-03-17T00:00:00.000Z',
    'ShippedDate': '1998-03-20T00:00:00.000Z',
    'Freight': 27.91,
    'ShipName': 'LINO-Delicatesses',
    'ShipAddress': 'Ave. 5 de Mayo Porlamar',
    'ShipCity': 'I. de Margarita',
    'ShipRegion': 'Nueva Esparta',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10955,
    'CustomerID': 'FOLKO',
    'OrderDate': '1998-03-17T00:00:00.000Z',
    'ShippedDate': '1998-03-20T00:00:00.000Z',
    'Freight': 3.26,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10956,
    'CustomerID': 'BLAUS',
    'OrderDate': '1998-03-17T00:00:00.000Z',
    'ShippedDate': '1998-03-20T00:00:00.000Z',
    'Freight': 44.65,
    'ShipName': 'Blauer See Delikatessen',
    'ShipAddress': 'Forsterstr. 57',
    'ShipCity': 'Mannheim',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  }

```

```

},
{
  'OrderID': 10957,
  'CustomerID': 'HILAA',
  'OrderDate': '1998-03-18T00:00:00.000Z',
  'ShippedDate': '1998-03-27T00:00:00.000Z',
  'Freight': 105.36,
  'ShipName': 'HILARION-Abastos',
  'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
  'ShipCity': 'San Cristóbal',
  'ShipRegion': 'Táchira',
  'ShipCountry': 'Venezuela'
},
{
  'OrderID': 10958,
  'CustomerID': 'OCEAN',
  'OrderDate': '1998-03-18T00:00:00.000Z',
  'ShippedDate': '1998-03-27T00:00:00.000Z',
  'Freight': 49.56,
  'ShipName': 'Océano Atlántico Ltda.',
  'ShipAddress': 'Ing. Gustavo Moncada 8585 Piso 20-A',
  'ShipCity': 'Buenos Aires',
  'ShipRegion': null,
  'ShipCountry': 'Argentina'
},
{
  'OrderID': 10959,
  'CustomerID': 'GOURL',
  'OrderDate': '1998-03-18T00:00:00.000Z',
  'ShippedDate': '1998-03-23T00:00:00.000Z',
  'Freight': 4.98,
  'ShipName': 'Gourmet Lanchonetes',
  'ShipAddress': 'Av. Brasil, 442',
  'ShipCity': 'Campinas',
  'ShipRegion': 'SP',
  'ShipCountry': 'Brazil'
},
{
  'OrderID': 10960,
  'CustomerID': 'HILAA',
  'OrderDate': '1998-03-19T00:00:00.000Z',
  'ShippedDate': '1998-04-08T00:00:00.000Z',
  'Freight': 2.08,
  'ShipName': 'HILARION-Abastos',
  'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
  'ShipCity': 'San Cristóbal',
  'ShipRegion': 'Táchira',
  'ShipCountry': 'Venezuela'
},
{
  'OrderID': 10961,
  'CustomerID': 'QUEEN',
  'OrderDate': '1998-03-19T00:00:00.000Z',
  'ShippedDate': '1998-03-30T00:00:00.000Z',
  'Freight': 104.47,
  'ShipName': 'Queen Cozinha',
  'ShipAddress': 'Alameda dos Canários, 891',

```

```

    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10962,
    'CustomerID': 'QUICK',
    'OrderDate': '1998-03-19T00:00:00.000Z',
    'ShippedDate': '1998-03-23T00:00:00.000Z',
    'Freight': 275.79,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10963,
    'CustomerID': 'FURIB',
    'OrderDate': '1998-03-19T00:00:00.000Z',
    'ShippedDate': '1998-03-26T00:00:00.000Z',
    'Freight': 2.7,
    'ShipName': 'Furia Bacalhau e Frutos do Mar',
    'ShipAddress': 'Jardim das rosas n. 32',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,
    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 10964,
    'CustomerID': 'SPECB',
    'OrderDate': '1998-03-20T00:00:00.000Z',
    'ShippedDate': '1998-03-24T00:00:00.000Z',
    'Freight': 87.38,
    'ShipName': 'Spécialités du monde',
    'ShipAddress': '25, rue Lauriston',
    'ShipCity': 'Paris',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10965,
    'CustomerID': 'OLDWO',
    'OrderDate': '1998-03-20T00:00:00.000Z',
    'ShippedDate': '1998-03-30T00:00:00.000Z',
    'Freight': 144.38,
    'ShipName': 'Old World Delicatessen',
    'ShipAddress': '2743 Bering St.',
    'ShipCity': 'Anchorage',
    'ShipRegion': 'AK',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10966,
    'CustomerID': 'CHOPS',
    'OrderDate': '1998-03-20T00:00:00.000Z',
    'ShippedDate': '1998-04-08T00:00:00.000Z',

```

```

    'Freight': 27.19,
    'ShipName': 'Chop-suey Chinese',
    'ShipAddress': 'Hauptstr. 31',
    'ShipCity': 'Bern',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10967,
    'CustomerID': 'TOMSP',
    'OrderDate': '1998-03-23T00:00:00.000Z',
    'ShippedDate': '1998-04-02T00:00:00.000Z',
    'Freight': 62.22,
    'ShipName': 'Toms Spezialitäten',
    'ShipAddress': 'Luisenstr. 48',
    'ShipCity': 'Münster',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10968,
    'CustomerID': 'ERNSH',
    'OrderDate': '1998-03-23T00:00:00.000Z',
    'ShippedDate': '1998-04-01T00:00:00.000Z',
    'Freight': 74.6,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10969,
    'CustomerID': 'COMMI',
    'OrderDate': '1998-03-23T00:00:00.000Z',
    'ShippedDate': '1998-03-30T00:00:00.000Z',
    'Freight': 0.21,
    'ShipName': 'Comércio Mineiro',
    'ShipAddress': 'Av. dos Lusíadas, 23',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10970,
    'CustomerID': 'BOLID',
    'OrderDate': '1998-03-24T00:00:00.000Z',
    'ShippedDate': '1998-04-24T00:00:00.000Z',
    'Freight': 16.16,
    'ShipName': 'Bólido Comidas preparadas',
    'ShipAddress': 'C/ Araquil, 67',
    'ShipCity': 'Madrid',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10971,

```

```

    'CustomerID': 'FRANR',
    'OrderDate': '1998-03-24T00:00:00.000Z',
    'ShippedDate': '1998-04-02T00:00:00.000Z',
    'Freight': 121.82,
    'ShipName': 'France restauration',
    'ShipAddress': '54, rue Royale',
    'ShipCity': 'Nantes',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10972,
    'CustomerID': 'LACOR',
    'OrderDate': '1998-03-24T00:00:00.000Z',
    'ShippedDate': '1998-03-26T00:00:00.000Z',
    'Freight': 0.02,
    'ShipName': 'La corne d\'abondance',
    'ShipAddress': '67, avenue de l\'Europe',
    'ShipCity': 'Versailles',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10973,
    'CustomerID': 'LACOR',
    'OrderDate': '1998-03-24T00:00:00.000Z',
    'ShippedDate': '1998-03-27T00:00:00.000Z',
    'Freight': 15.17,
    'ShipName': 'La corne d\'abondance',
    'ShipAddress': '67, avenue de l\'Europe',
    'ShipCity': 'Versailles',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10974,
    'CustomerID': 'SPLIR',
    'OrderDate': '1998-03-25T00:00:00.000Z',
    'ShippedDate': '1998-04-03T00:00:00.000Z',
    'Freight': 12.96,
    'ShipName': 'Split Rail Beer & Ale',
    'ShipAddress': 'P.O. Box 555',
    'ShipCity': 'Lander',
    'ShipRegion': 'WY',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10975,
    'CustomerID': 'BOTTM',
    'OrderDate': '1998-03-25T00:00:00.000Z',
    'ShippedDate': '1998-03-27T00:00:00.000Z',
    'Freight': 32.27,
    'ShipName': 'Bottom-Dollar Markets',
    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  }

```

```

    },
    {
        'OrderID': 10976,
        'CustomerID': 'HILAA',
        'OrderDate': '1998-03-25T00:00:00.000Z',
        'ShippedDate': '1998-04-03T00:00:00.000Z',
        'Freight': 37.97,
        'ShipName': 'HILARION-Abastos',
        'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
        'ShipCity': 'San Cristóbal',
        'ShipRegion': 'Táchira',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10977,
        'CustomerID': 'FOLKO',
        'OrderDate': '1998-03-26T00:00:00.000Z',
        'ShippedDate': '1998-04-10T00:00:00.000Z',
        'Freight': 208.5,
        'ShipName': 'Folk och fä HB',
        'ShipAddress': 'Åkergatan 24',
        'ShipCity': 'Bräcke',
        'ShipRegion': null,
        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 10978,
        'CustomerID': 'MAISD',
        'OrderDate': '1998-03-26T00:00:00.000Z',
        'ShippedDate': '1998-04-23T00:00:00.000Z',
        'Freight': 32.82,
        'ShipName': 'Maison Dewey',
        'ShipAddress': 'Rue Joseph-Bens 532',
        'ShipCity': 'Bruxelles',
        'ShipRegion': null,
        'ShipCountry': 'Belgium'
    },
    {
        'OrderID': 10979,
        'CustomerID': 'ERNSH',
        'OrderDate': '1998-03-26T00:00:00.000Z',
        'ShippedDate': '1998-03-31T00:00:00.000Z',
        'Freight': 353.07,
        'ShipName': 'Ernst Handel',
        'ShipAddress': 'Kirchgasse 6',
        'ShipCity': 'Graz',
        'ShipRegion': null,
        'ShipCountry': 'Austria'
    },
    {
        'OrderID': 10980,
        'CustomerID': 'FOLKO',
        'OrderDate': '1998-03-27T00:00:00.000Z',
        'ShippedDate': '1998-04-17T00:00:00.000Z',
        'Freight': 1.26,
        'ShipName': 'Folk och fä HB',
        'ShipAddress': 'Åkergatan 24',

```

```

        'ShipCity': 'Bräcke',
        'ShipRegion': null,
        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 10981,
        'CustomerID': 'HANAR',
        'OrderDate': '1998-03-27T00:00:00.000Z',
        'ShippedDate': '1998-04-02T00:00:00.000Z',
        'Freight': 193.37,
        'ShipName': 'Hanari Carnes',
        'ShipAddress': 'Rua do Paço, 67',
        'ShipCity': 'Rio de Janeiro',
        'ShipRegion': 'RJ',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10982,
        'CustomerID': 'BOTTM',
        'OrderDate': '1998-03-27T00:00:00.000Z',
        'ShippedDate': '1998-04-08T00:00:00.000Z',
        'Freight': 14.01,
        'ShipName': 'Bottom-Dollar Markets',
        'ShipAddress': '23 Tsawassen Blvd.',
        'ShipCity': 'Tsawassen',
        'ShipRegion': 'BC',
        'ShipCountry': 'Canada'
    },
    {
        'OrderID': 10983,
        'CustomerID': 'SAVEA',
        'OrderDate': '1998-03-27T00:00:00.000Z',
        'ShippedDate': '1998-04-06T00:00:00.000Z',
        'Freight': 657.54,
        'ShipName': 'Save-a-lot Markets',
        'ShipAddress': '187 Suffolk Ln.',
        'ShipCity': 'Boise',
        'ShipRegion': 'ID',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10984,
        'CustomerID': 'SAVEA',
        'OrderDate': '1998-03-30T00:00:00.000Z',
        'ShippedDate': '1998-04-03T00:00:00.000Z',
        'Freight': 211.22,
        'ShipName': 'Save-a-lot Markets',
        'ShipAddress': '187 Suffolk Ln.',
        'ShipCity': 'Boise',
        'ShipRegion': 'ID',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10985,
        'CustomerID': 'HUNGO',
        'OrderDate': '1998-03-30T00:00:00.000Z',
        'ShippedDate': '1998-04-02T00:00:00.000Z',

```



```

    'Freight': 91.51,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10986,
    'CustomerID': 'OCEAN',
    'OrderDate': '1998-03-30T00:00:00.000Z',
    'ShippedDate': '1998-04-21T00:00:00.000Z',
    'Freight': 217.86,
    'ShipName': 'Océano Atlántico Ltda.',
    'ShipAddress': 'Ing. Gustavo Moncada 8585 Piso 20-A',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 10987,
    'CustomerID': 'EASTC',
    'OrderDate': '1998-03-31T00:00:00.000Z',
    'ShippedDate': '1998-04-06T00:00:00.000Z',
    'Freight': 185.48,
    'ShipName': 'Eastern Connection',
    'ShipAddress': '35 King George',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10988,
    'CustomerID': 'RATTC',
    'OrderDate': '1998-03-31T00:00:00.000Z',
    'ShippedDate': '1998-04-10T00:00:00.000Z',
    'Freight': 61.14,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10989,
    'CustomerID': 'QUEDE',
    'OrderDate': '1998-03-31T00:00:00.000Z',
    'ShippedDate': '1998-04-02T00:00:00.000Z',
    'Freight': 34.76,
    'ShipName': 'Que Delícia',
    'ShipAddress': 'Rua da Panificadora, 12',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10990,

```

```

    'CustomerID': 'ERNSH',
    'OrderDate': '1998-04-01T00:00:00.000Z',
    'ShippedDate': '1998-04-07T00:00:00.000Z',
    'Freight': 117.61,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10991,
    'CustomerID': 'QUICK',
    'OrderDate': '1998-04-01T00:00:00.000Z',
    'ShippedDate': '1998-04-07T00:00:00.000Z',
    'Freight': 38.51,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10992,
    'CustomerID': 'THEBI',
    'OrderDate': '1998-04-01T00:00:00.000Z',
    'ShippedDate': '1998-04-03T00:00:00.000Z',
    'Freight': 4.27,
    'ShipName': 'The Big Cheese',
    'ShipAddress': '89 Jefferson Way Suite 2',
    'ShipCity': 'Portland',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10993,
    'CustomerID': 'FOLKO',
    'OrderDate': '1998-04-01T00:00:00.000Z',
    'ShippedDate': '1998-04-10T00:00:00.000Z',
    'Freight': 8.81,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10994,
    'CustomerID': 'VAFFE',
    'OrderDate': '1998-04-02T00:00:00.000Z',
    'ShippedDate': '1998-04-09T00:00:00.000Z',
    'Freight': 65.53,
    'ShipName': 'Vaffeljernet',
    'ShipAddress': 'Smagsloget 45',
    'ShipCity': 'Århus',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  }

```

```

    },
    {
        'OrderID': 10995,
        'CustomerID': 'PERIC',
        'OrderDate': '1998-04-02T00:00:00.000Z',
        'ShippedDate': '1998-04-06T00:00:00.000Z',
        'Freight': 46,
        'ShipName': 'Pericles Comidas clásicas',
        'ShipAddress': 'Calle Dr. Jorge Cash 321',
        'ShipCity': 'México D.F.',
        'ShipRegion': null,
        'ShipCountry': 'Mexico'
    },
    {
        'OrderID': 10996,
        'CustomerID': 'QUICK',
        'OrderDate': '1998-04-02T00:00:00.000Z',
        'ShippedDate': '1998-04-10T00:00:00.000Z',
        'Freight': 1.12,
        'ShipName': 'QUICK-Stop',
        'ShipAddress': 'Taucherstraße 10',
        'ShipCity': 'Cunewalde',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10997,
        'CustomerID': 'LILAS',
        'OrderDate': '1998-04-03T00:00:00.000Z',
        'ShippedDate': '1998-04-13T00:00:00.000Z',
        'Freight': 73.91,
        'ShipName': 'LILA-Supermercado',
        'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
        'ShipCity': 'Barquisimeto',
        'ShipRegion': 'Lara',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10998,
        'CustomerID': 'WOLZA',
        'OrderDate': '1998-04-03T00:00:00.000Z',
        'ShippedDate': '1998-04-17T00:00:00.000Z',
        'Freight': 20.31,
        'ShipName': 'Wolski Zajazd',
        'ShipAddress': 'ul. Filtrowa 68',
        'ShipCity': 'Warszawa',
        'ShipRegion': null,
        'ShipCountry': 'Poland'
    },
    {
        'OrderID': 10999,
        'CustomerID': 'OTTIK',
        'OrderDate': '1998-04-03T00:00:00.000Z',
        'ShippedDate': '1998-04-10T00:00:00.000Z',
        'Freight': 96.35,
        'ShipName': 'Ottilies Käseladen',
        'ShipAddress': 'Mehrheimerstr. 369',
    }

```

```

    'ShipCity': 'Köln',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 11000,
    'CustomerID': 'RATTC',
    'OrderDate': '1998-04-06T00:00:00.000Z',
    'ShippedDate': '1998-04-14T00:00:00.000Z',
    'Freight': 55.12,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 11001,
    'CustomerID': 'FOLKO',
    'OrderDate': '1998-04-06T00:00:00.000Z',
    'ShippedDate': '1998-04-14T00:00:00.000Z',
    'Freight': 197.3,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 11002,
    'CustomerID': 'SAVEA',
    'OrderDate': '1998-04-06T00:00:00.000Z',
    'ShippedDate': '1998-04-16T00:00:00.000Z',
    'Freight': 141.16,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 11003,
    'CustomerID': 'THECR',
    'OrderDate': '1998-04-06T00:00:00.000Z',
    'ShippedDate': '1998-04-08T00:00:00.000Z',
    'Freight': 14.91,
    'ShipName': 'The Cracker Box',
    'ShipAddress': '55 Grizzly Peak Rd.',
    'ShipCity': 'Butte',
    'ShipRegion': 'MT',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 11004,
    'CustomerID': 'MAISD',
    'OrderDate': '1998-04-07T00:00:00.000Z',
    'ShippedDate': '1998-04-20T00:00:00.000Z',

```

```

    'Freight': 44.84,
    'ShipName': 'Maison Dewey',
    'ShipAddress': 'Rue Joseph-Bens 532',
    'ShipCity': 'Bruxelles',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 11005,
    'CustomerID': 'WILMK',
    'OrderDate': '1998-04-07T00:00:00.000Z',
    'ShippedDate': '1998-04-10T00:00:00.000Z',
    'Freight': 0.75,
    'ShipName': 'Wilman Kala',
    'ShipAddress': 'Keskuskatu 45',
    'ShipCity': 'Helsinki',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 11006,
    'CustomerID': 'GREAL',
    'OrderDate': '1998-04-07T00:00:00.000Z',
    'ShippedDate': '1998-04-15T00:00:00.000Z',
    'Freight': 25.19,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 11007,
    'CustomerID': 'PRINI',
    'OrderDate': '1998-04-08T00:00:00.000Z',
    'ShippedDate': '1998-04-13T00:00:00.000Z',
    'Freight': 202.24,
    'ShipName': 'Princesa Isabel Vinhos',
    'ShipAddress': 'Estrada da saúde n. 58',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,
    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 11008,
    'CustomerID': 'ERNSH',
    'OrderDate': '1998-04-08T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 79.46,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 11009,

```

```

    'CustomerID': 'GODOS',
    'OrderDate': '1998-04-08T00:00:00.000Z',
    'ShippedDate': '1998-04-10T00:00:00.000Z',
    'Freight': 59.11,
    'ShipName': 'Godos Cocina Típica',
    'ShipAddress': 'C/ Romero, 33',
    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 11010,
    'CustomerID': 'REGGC',
    'OrderDate': '1998-04-09T00:00:00.000Z',
    'ShippedDate': '1998-04-21T00:00:00.000Z',
    'Freight': 28.71,
    'ShipName': 'Reggiani Caseifici',
    'ShipAddress': 'Strada Provinciale 124',
    'ShipCity': 'Reggio Emilia',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 11011,
    'CustomerID': 'ALFKI',
    'OrderDate': '1998-04-09T00:00:00.000Z',
    'ShippedDate': '1998-04-13T00:00:00.000Z',
    'Freight': 1.21,
    'ShipName': 'Alfred\ Futterkiste',
    'ShipAddress': 'Obere Str. 57',
    'ShipCity': 'Berlin',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 11012,
    'CustomerID': 'FRANK',
    'OrderDate': '1998-04-09T00:00:00.000Z',
    'ShippedDate': '1998-04-17T00:00:00.000Z',
    'Freight': 242.95,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 11013,
    'CustomerID': 'ROMEY',
    'OrderDate': '1998-04-09T00:00:00.000Z',
    'ShippedDate': '1998-04-10T00:00:00.000Z',
    'Freight': 32.99,
    'ShipName': 'Romero y tomillo',
    'ShipAddress': 'Gran Vía, 1',
    'ShipCity': 'Madrid',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  }

```

```

},
{
  'OrderID': 11014,
  'CustomerID': 'LINOD',
  'OrderDate': '1998-04-10T00:00:00.000Z',
  'ShippedDate': '1998-04-15T00:00:00.000Z',
  'Freight': 23.6,
  'ShipName': 'LINO-Delicateses',
  'ShipAddress': 'Ave. 5 de Mayo Porlamar',
  'ShipCity': 'I. de Margarita',
  'ShipRegion': 'Nueva Esparta',
  'ShipCountry': 'Venezuela'
},
{
  'OrderID': 11015,
  'CustomerID': 'SANTG',
  'OrderDate': '1998-04-10T00:00:00.000Z',
  'ShippedDate': '1998-04-20T00:00:00.000Z',
  'Freight': 4.62,
  'ShipName': 'Santé Gourmet',
  'ShipAddress': 'Erling Skakkess gate 78',
  'ShipCity': 'Stavern',
  'ShipRegion': null,
  'ShipCountry': 'Norway'
},
{
  'OrderID': 11016,
  'CustomerID': 'AROUT',
  'OrderDate': '1998-04-10T00:00:00.000Z',
  'ShippedDate': '1998-04-13T00:00:00.000Z',
  'Freight': 33.8,
  'ShipName': 'Around the Horn',
  'ShipAddress': 'Brook Farm Stratford St. Mary',
  'ShipCity': 'Colchester',
  'ShipRegion': 'Essex',
  'ShipCountry': 'UK'
},
{
  'OrderID': 11017,
  'CustomerID': 'ERNSH',
  'OrderDate': '1998-04-13T00:00:00.000Z',
  'ShippedDate': '1998-04-20T00:00:00.000Z',
  'Freight': 754.26,
  'ShipName': 'Ernst Handel',
  'ShipAddress': 'Kirchgasse 6',
  'ShipCity': 'Graz',
  'ShipRegion': null,
  'ShipCountry': 'Austria'
},
{
  'OrderID': 11018,
  'CustomerID': 'LONEP',
  'OrderDate': '1998-04-13T00:00:00.000Z',
  'ShippedDate': '1998-04-16T00:00:00.000Z',
  'Freight': 11.65,
  'ShipName': 'Lonesome Pine Restaurant',
  'ShipAddress': '89 Chiaroscuro Rd.',

```

```

    'ShipCity': 'Portland',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 11019,
    'CustomerID': 'RANCH',
    'OrderDate': '1998-04-13T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 3.17,
    'ShipName': 'Rancho grande',
    'ShipAddress': 'Av. del Libertador 900',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 11020,
    'CustomerID': 'OTTIK',
    'OrderDate': '1998-04-14T00:00:00.000Z',
    'ShippedDate': '1998-04-16T00:00:00.000Z',
    'Freight': 43.3,
    'ShipName': 'Ottilies Käseladen',
    'ShipAddress': 'Mehrheimerstr. 369',
    'ShipCity': 'Köln',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 11021,
    'CustomerID': 'QUICK',
    'OrderDate': '1998-04-14T00:00:00.000Z',
    'ShippedDate': '1998-04-21T00:00:00.000Z',
    'Freight': 297.18,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 11022,
    'CustomerID': 'HANAR',
    'OrderDate': '1998-04-14T00:00:00.000Z',
    'ShippedDate': '1998-05-04T00:00:00.000Z',
    'Freight': 6.27,
    'ShipName': 'Hanari Carnes',
    'ShipAddress': 'Rua do Paço, 67',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 11023,
    'CustomerID': 'BSBEV',
    'OrderDate': '1998-04-14T00:00:00.000Z',
    'ShippedDate': '1998-04-24T00:00:00.000Z',

```



```

    'Freight': 123.83,
    'ShipName': 'B\' Beverages',
    'ShipAddress': 'Fauntleroy Circus',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 11024,
    'CustomerID': 'EASTC',
    'OrderDate': '1998-04-15T00:00:00.000Z',
    'ShippedDate': '1998-04-20T00:00:00.000Z',
    'Freight': 74.36,
    'ShipName': 'Eastern Connection',
    'ShipAddress': '35 King George',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 11025,
    'CustomerID': 'WARTH',
    'OrderDate': '1998-04-15T00:00:00.000Z',
    'ShippedDate': '1998-04-24T00:00:00.000Z',
    'Freight': 29.17,
    'ShipName': 'Wartian Herkku',
    'ShipAddress': 'Torikatu 38',
    'ShipCity': 'Oulu',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 11026,
    'CustomerID': 'FRANS',
    'OrderDate': '1998-04-15T00:00:00.000Z',
    'ShippedDate': '1998-04-28T00:00:00.000Z',
    'Freight': 47.09,
    'ShipName': 'Franchi S.p.A.',
    'ShipAddress': 'Via Monte Bianco 34',
    'ShipCity': 'Torino',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 11027,
    'CustomerID': 'BOTTM',
    'OrderDate': '1998-04-16T00:00:00.000Z',
    'ShippedDate': '1998-04-20T00:00:00.000Z',
    'Freight': 52.52,
    'ShipName': 'Bottom-Dollar Markets',
    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 11028,

```

```

      'CustomerID': 'KOENE',
      'OrderDate': '1998-04-16T00:00:00.000Z',
      'ShippedDate': '1998-04-22T00:00:00.000Z',
      'Freight': 29.59,
      'ShipName': 'Königlich Essen',
      'ShipAddress': 'Maubelstr. 90',
      'ShipCity': 'Brandenburg',
      'ShipRegion': null,
      'ShipCountry': 'Germany'
    },
    {
      'OrderID': 11029,
      'CustomerID': 'CHOPS',
      'OrderDate': '1998-04-16T00:00:00.000Z',
      'ShippedDate': '1998-04-27T00:00:00.000Z',
      'Freight': 47.84,
      'ShipName': 'Chop-suey Chinese',
      'ShipAddress': 'Hauptstr. 31',
      'ShipCity': 'Bern',
      'ShipRegion': null,
      'ShipCountry': 'Switzerland'
    },
    {
      'OrderID': 11030,
      'CustomerID': 'SAVEA',
      'OrderDate': '1998-04-17T00:00:00.000Z',
      'ShippedDate': '1998-04-27T00:00:00.000Z',
      'Freight': 830.75,
      'ShipName': 'Save-a-lot Markets',
      'ShipAddress': '187 Suffolk Ln.',
      'ShipCity': 'Boise',
      'ShipRegion': 'ID',
      'ShipCountry': 'USA'
    },
    {
      'OrderID': 11031,
      'CustomerID': 'SAVEA',
      'OrderDate': '1998-04-17T00:00:00.000Z',
      'ShippedDate': '1998-04-24T00:00:00.000Z',
      'Freight': 227.22,
      'ShipName': 'Save-a-lot Markets',
      'ShipAddress': '187 Suffolk Ln.',
      'ShipCity': 'Boise',
      'ShipRegion': 'ID',
      'ShipCountry': 'USA'
    },
    {
      'OrderID': 11032,
      'CustomerID': 'WHITC',
      'OrderDate': '1998-04-17T00:00:00.000Z',
      'ShippedDate': '1998-04-23T00:00:00.000Z',
      'Freight': 606.19,
      'ShipName': 'White Clover Markets',
      'ShipAddress': '1029 - 12th Ave. S.',
      'ShipCity': 'Seattle',
      'ShipRegion': 'WA',
      'ShipCountry': 'USA'
    }
  ]

```

```

},
{
  'OrderID': 11033,
  'CustomerID': 'RICSU',
  'OrderDate': '1998-04-17T00:00:00.000Z',
  'ShippedDate': '1998-04-23T00:00:00.000Z',
  'Freight': 84.74,
  'ShipName': 'Richter Supermarkt',
  'ShipAddress': 'Starenweg 5',
  'ShipCity': 'Genève',
  'ShipRegion': null,
  'ShipCountry': 'Switzerland'
},
{
  'OrderID': 11034,
  'CustomerID': 'OLDWO',
  'OrderDate': '1998-04-20T00:00:00.000Z',
  'ShippedDate': '1998-04-27T00:00:00.000Z',
  'Freight': 40.32,
  'ShipName': 'Old World Delicatessen',
  'ShipAddress': '2743 Bering St.',
  'ShipCity': 'Anchorage',
  'ShipRegion': 'AK',
  'ShipCountry': 'USA'
},
{
  'OrderID': 11035,
  'CustomerID': 'SUPRD',
  'OrderDate': '1998-04-20T00:00:00.000Z',
  'ShippedDate': '1998-04-24T00:00:00.000Z',
  'Freight': 0.17,
  'ShipName': 'Suprêmes délices',
  'ShipAddress': 'Boulevard Tirou, 255',
  'ShipCity': 'Charleroi',
  'ShipRegion': null,
  'ShipCountry': 'Belgium'
},
{
  'OrderID': 11036,
  'CustomerID': 'DRACD',
  'OrderDate': '1998-04-20T00:00:00.000Z',
  'ShippedDate': '1998-04-22T00:00:00.000Z',
  'Freight': 149.47,
  'ShipName': 'Drachenblut Delikatessen',
  'ShipAddress': 'Walserweg 21',
  'ShipCity': 'Aachen',
  'ShipRegion': null,
  'ShipCountry': 'Germany'
},
{
  'OrderID': 11037,
  'CustomerID': 'GODOS',
  'OrderDate': '1998-04-21T00:00:00.000Z',
  'ShippedDate': '1998-04-27T00:00:00.000Z',
  'Freight': 3.2,
  'ShipName': 'Godos Cocina Típica',
  'ShipAddress': 'C/ Romero, 33',

```

```

    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 11038,
    'CustomerID': 'SUPRD',
    'OrderDate': '1998-04-21T00:00:00.000Z',
    'ShippedDate': '1998-04-30T00:00:00.000Z',
    'Freight': 29.59,
    'ShipName': 'Suprêmes délices',
    'ShipAddress': 'Boulevard Tirou, 255',
    'ShipCity': 'Charleroi',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 11039,
    'CustomerID': 'LINOD',
    'OrderDate': '1998-04-21T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 65,
    'ShipName': 'LINO-Delicatesses',
    'ShipAddress': 'Ave. 5 de Mayo Porlamar',
    'ShipCity': 'I. de Margarita',
    'ShipRegion': 'Nueva Esparta',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 11040,
    'CustomerID': 'GREAL',
    'OrderDate': '1998-04-22T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 18.84,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 11041,
    'CustomerID': 'CHOPS',
    'OrderDate': '1998-04-22T00:00:00.000Z',
    'ShippedDate': '1998-04-28T00:00:00.000Z',
    'Freight': 48.22,
    'ShipName': 'Chop-suey Chinese',
    'ShipAddress': 'Hauptstr. 31',
    'ShipCity': 'Bern',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 11042,
    'CustomerID': 'COMMI',
    'OrderDate': '1998-04-22T00:00:00.000Z',
    'ShippedDate': '1998-05-01T00:00:00.000Z',

```

```

    'Freight': 29.99,
    'ShipName': 'Comércio Mineiro',
    'ShipAddress': 'Av. dos Lusíadas, 23',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 11043,
    'CustomerID': 'SPECB',
    'OrderDate': '1998-04-22T00:00:00.000Z',
    'ShippedDate': '1998-04-29T00:00:00.000Z',
    'Freight': 8.8,
    'ShipName': 'Spécialités du monde',
    'ShipAddress': '25, rue Lauriston',
    'ShipCity': 'Paris',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 11044,
    'CustomerID': 'WOLZA',
    'OrderDate': '1998-04-23T00:00:00.000Z',
    'ShippedDate': '1998-05-01T00:00:00.000Z',
    'Freight': 8.72,
    'ShipName': 'Wolski Zajazd',
    'ShipAddress': 'ul. Filtrowa 68',
    'ShipCity': 'Warszawa',
    'ShipRegion': null,
    'ShipCountry': 'Poland'
  },
  {
    'OrderID': 11045,
    'CustomerID': 'BOTTM',
    'OrderDate': '1998-04-23T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 70.58,
    'ShipName': 'Bottom-Dollar Markets',
    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 11046,
    'CustomerID': 'WANDK',
    'OrderDate': '1998-04-23T00:00:00.000Z',
    'ShippedDate': '1998-04-24T00:00:00.000Z',
    'Freight': 71.64,
    'ShipName': 'Die Wandernde Kuh',
    'ShipAddress': 'Adenauerallee 900',
    'ShipCity': 'Stuttgart',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 11047,

```

```

    'CustomerID': 'EASTC',
    'OrderDate': '1998-04-24T00:00:00.000Z',
    'ShippedDate': '1998-05-01T00:00:00.000Z',
    'Freight': 46.62,
    'ShipName': 'Eastern Connection',
    'ShipAddress': '35 King George',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 11048,
    'CustomerID': 'BOTTM',
    'OrderDate': '1998-04-24T00:00:00.000Z',
    'ShippedDate': '1998-04-30T00:00:00.000Z',
    'Freight': 24.12,
    'ShipName': 'Bottom-Dollar Markets',
    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 11049,
    'CustomerID': 'GOURL',
    'OrderDate': '1998-04-24T00:00:00.000Z',
    'ShippedDate': '1998-05-04T00:00:00.000Z',
    'Freight': 8.34,
    'ShipName': 'Gourmet Lanchonetes',
    'ShipAddress': 'Av. Brasil, 442',
    'ShipCity': 'Campinas',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 11050,
    'CustomerID': 'FOLKO',
    'OrderDate': '1998-04-27T00:00:00.000Z',
    'ShippedDate': '1998-05-05T00:00:00.000Z',
    'Freight': 59.41,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 11051,
    'CustomerID': 'LAMAI',
    'OrderDate': '1998-04-27T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 2.79,
    'ShipName': 'La maison d\'Asie',
    'ShipAddress': '1 rue Alsace-Lorraine',
    'ShipCity': 'Toulouse',
    'ShipRegion': null,
    'ShipCountry': 'France'
  }

```

```

},
{
  'OrderID': 11052,
  'CustomerID': 'HANAR',
  'OrderDate': '1998-04-27T00:00:00.000Z',
  'ShippedDate': '1998-05-01T00:00:00.000Z',
  'Freight': 67.26,
  'ShipName': 'Hanari Carnes',
  'ShipAddress': 'Rua do Paço, 67',
  'ShipCity': 'Rio de Janeiro',
  'ShipRegion': 'RJ',
  'ShipCountry': 'Brazil'
},
{
  'OrderID': 11053,
  'CustomerID': 'PICCO',
  'OrderDate': '1998-04-27T00:00:00.000Z',
  'ShippedDate': '1998-04-29T00:00:00.000Z',
  'Freight': 53.05,
  'ShipName': 'Piccolo und mehr',
  'ShipAddress': 'Geislweg 14',
  'ShipCity': 'Salzburg',
  'ShipRegion': null,
  'ShipCountry': 'Austria'
},
{
  'OrderID': 11054,
  'CustomerID': 'CACTU',
  'OrderDate': '1998-04-28T00:00:00.000Z',
  'ShippedDate': null,
  'Freight': 0.33,
  'ShipName': 'Cactus Comidas para llevar',
  'ShipAddress': 'Cerrito 333',
  'ShipCity': 'Buenos Aires',
  'ShipRegion': null,
  'ShipCountry': 'Argentina'
},
{
  'OrderID': 11055,
  'CustomerID': 'HILAA',
  'OrderDate': '1998-04-28T00:00:00.000Z',
  'ShippedDate': '1998-05-05T00:00:00.000Z',
  'Freight': 120.92,
  'ShipName': 'HILARION-Abastos',
  'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
  'ShipCity': 'San Cristóbal',
  'ShipRegion': 'Táchira',
  'ShipCountry': 'Venezuela'
},
{
  'OrderID': 11056,
  'CustomerID': 'EASTC',
  'OrderDate': '1998-04-28T00:00:00.000Z',
  'ShippedDate': '1998-05-01T00:00:00.000Z',
  'Freight': 278.96,
  'ShipName': 'Eastern Connection',
  'ShipAddress': '35 King George',

```

```

        'ShipCity': 'London',
        'ShipRegion': null,
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 11057,
        'CustomerID': 'NORTS',
        'OrderDate': '1998-04-29T00:00:00.000Z',
        'ShippedDate': '1998-05-01T00:00:00.000Z',
        'Freight': 4.13,
        'ShipName': 'North/South',
        'ShipAddress': 'South House 300 Queensbridge',
        'ShipCity': 'London',
        'ShipRegion': null,
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 11058,
        'CustomerID': 'BLAUS',
        'OrderDate': '1998-04-29T00:00:00.000Z',
        'ShippedDate': null,
        'Freight': 31.14,
        'ShipName': 'Blauer See Delikatessen',
        'ShipAddress': 'Forsterstr. 57',
        'ShipCity': 'Mannheim',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 11059,
        'CustomerID': 'RICAR',
        'OrderDate': '1998-04-29T00:00:00.000Z',
        'ShippedDate': null,
        'Freight': 85.8,
        'ShipName': 'Ricardo Adocicados',
        'ShipAddress': 'Av. Copacabana, 267',
        'ShipCity': 'Rio de Janeiro',
        'ShipRegion': 'RJ',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 11060,
        'CustomerID': 'FRANS',
        'OrderDate': '1998-04-30T00:00:00.000Z',
        'ShippedDate': '1998-05-04T00:00:00.000Z',
        'Freight': 10.98,
        'ShipName': 'Franchi S.p.A.',
        'ShipAddress': 'Via Monte Bianco 34',
        'ShipCity': 'Torino',
        'ShipRegion': null,
        'ShipCountry': 'Italy'
    },
    {
        'OrderID': 11061,
        'CustomerID': 'GREAL',
        'OrderDate': '1998-04-30T00:00:00.000Z',
        'ShippedDate': null,

```



```

    'Freight': 14.01,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 11062,
    'CustomerID': 'REGGC',
    'OrderDate': '1998-04-30T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 29.93,
    'ShipName': 'Reggiani Caseifici',
    'ShipAddress': 'Strada Provinciale 124',
    'ShipCity': 'Reggio Emilia',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 11063,
    'CustomerID': 'HUNGO',
    'OrderDate': '1998-04-30T00:00:00.000Z',
    'ShippedDate': '1998-05-06T00:00:00.000Z',
    'Freight': 81.73,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 11064,
    'CustomerID': 'SAVEA',
    'OrderDate': '1998-05-01T00:00:00.000Z',
    'ShippedDate': '1998-05-04T00:00:00.000Z',
    'Freight': 30.09,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 11065,
    'CustomerID': 'LILAS',
    'OrderDate': '1998-05-01T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 12.91,
    'ShipName': 'LILA-Supermercado',
    'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 11066,

```

```

      'CustomerID': 'WHITC',
      'OrderDate': '1998-05-01T00:00:00.000Z',
      'ShippedDate': '1998-05-04T00:00:00.000Z',
      'Freight': 44.72,
      'ShipName': 'White Clover Markets',
      'ShipAddress': '1029 - 12th Ave. S.',
      'ShipCity': 'Seattle',
      'ShipRegion': 'WA',
      'ShipCountry': 'USA'
    },
    {
      'OrderID': 11067,
      'CustomerID': 'DRACD',
      'OrderDate': '1998-05-04T00:00:00.000Z',
      'ShippedDate': '1998-05-06T00:00:00.000Z',
      'Freight': 7.98,
      'ShipName': 'Drachenblut Delikatessen',
      'ShipAddress': 'Walserweg 21',
      'ShipCity': 'Aachen',
      'ShipRegion': null,
      'ShipCountry': 'Germany'
    },
    {
      'OrderID': 11068,
      'CustomerID': 'QUEEN',
      'OrderDate': '1998-05-04T00:00:00.000Z',
      'ShippedDate': null,
      'Freight': 81.75,
      'ShipName': 'Queen Cozinha',
      'ShipAddress': 'Alameda dos Canários, 891',
      'ShipCity': 'Sao Paulo',
      'ShipRegion': 'SP',
      'ShipCountry': 'Brazil'
    },
    {
      'OrderID': 11069,
      'CustomerID': 'TORTU',
      'OrderDate': '1998-05-04T00:00:00.000Z',
      'ShippedDate': '1998-05-06T00:00:00.000Z',
      'Freight': 15.67,
      'ShipName': 'Tortuga Restaurante',
      'ShipAddress': 'Avda. Azteca 123',
      'ShipCity': 'México D.F.',
      'ShipRegion': null,
      'ShipCountry': 'Mexico'
    },
    {
      'OrderID': 11070,
      'CustomerID': 'LEHMS',
      'OrderDate': '1998-05-05T00:00:00.000Z',
      'ShippedDate': null,
      'Freight': 136,
      'ShipName': 'Lehmanns Marktstand',
      'ShipAddress': 'Magazinweg 7',
      'ShipCity': 'Frankfurt a.M.',
      'ShipRegion': null,
      'ShipCountry': 'Germany'
    }
  ]

```

```

},
{
  'OrderID': 11071,
  'CustomerID': 'LILAS',
  'OrderDate': '1998-05-05T00:00:00.000Z',
  'ShippedDate': null,
  'Freight': 0.93,
  'ShipName': 'LILA-Supermercado',
  'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
  'ShipCity': 'Barquisimeto',
  'ShipRegion': 'Lara',
  'ShipCountry': 'Venezuela'
},
{
  'OrderID': 11072,
  'CustomerID': 'ERNSH',
  'OrderDate': '1998-05-05T00:00:00.000Z',
  'ShippedDate': null,
  'Freight': 258.64,
  'ShipName': 'Ernst Handel',
  'ShipAddress': 'Kirchgasse 6',
  'ShipCity': 'Graz',
  'ShipRegion': null,
  'ShipCountry': 'Austria'
},
{
  'OrderID': 11073,
  'CustomerID': 'PERIC',
  'OrderDate': '1998-05-05T00:00:00.000Z',
  'ShippedDate': null,
  'Freight': 24.95,
  'ShipName': 'Pericles Comidas clásicas',
  'ShipAddress': 'Calle Dr. Jorge Cash 321',
  'ShipCity': 'México D.F.',
  'ShipRegion': null,
  'ShipCountry': 'Mexico'
},
{
  'OrderID': 11074,
  'CustomerID': 'SIMOB',
  'OrderDate': '1998-05-06T00:00:00.000Z',
  'ShippedDate': null,
  'Freight': 18.44,
  'ShipName': 'Simons bistro',
  'ShipAddress': 'Vinbæltet 34',
  'ShipCity': 'Kobenhavn',
  'ShipRegion': null,
  'ShipCountry': 'Denmark'
},
{
  'OrderID': 11075,
  'CustomerID': 'RICSU',
  'OrderDate': '1998-05-06T00:00:00.000Z',
  'ShippedDate': null,
  'Freight': 6.19,
  'ShipName': 'Richter Supermarkt',
  'ShipAddress': 'Starenweg 5',

```

```

        'ShipCity': 'Genève',
        'ShipRegion': null,
        'ShipCountry': 'Switzerland'
    },
    {
        'OrderID': 11076,
        'CustomerID': 'BONAP',
        'OrderDate': '1998-05-06T00:00:00.000Z',
        'ShippedDate': null,
        'Freight': 38.28,
        'ShipName': 'Bon app',
        'ShipAddress': '12, rue des Bouchers',
        'ShipCity': 'Marseille',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 11077,
        'CustomerID': 'RATTC',
        'OrderDate': '1998-05-06T00:00:00.000Z',
        'ShippedDate': null,
        'Freight': 8.53,
        'ShipName': 'Rattlesnake Canyon Grocery',
        'ShipAddress': '2817 Milton Dr.',
        'ShipCity': 'Albuquerque',
        'ShipRegion': 'NM',
        'ShipCountry': 'USA'
    }
]);
export const orderData = JSON.parse(stringData, (field, value) => {
    let dupValue = value;
    if (typeof value === 'string' && /^(\\d{4}\\-\\d\\d\\-\\d\\d([tT][\\d:\\.]*){1})([zZ]|([+\\-])(\\d\\d)?(\\d\\d)?$/).test(value)) {
        let arr = dupValue.split(/[^0-9]/);
        let arg = parseInt(arr[4], 10);
        let arg1 = parseInt(arr[5], 10);
        value = new Date(parseInt(arr[0], 10), parseInt(arr[1], 10) - 1,
            parseInt(arr[2], 10), parseInt(arr[3], 10), arg, arg1);
    }
    return value;
});
export const categoryData = [
    {
        'CategoryName': 'Beverages',
        'ProductName': 'Chai',
        'QuantityPerUnit': '10 boxes x 20 bags',
        'UnitsInStock': 39,
        'Discontinued': true
    },
    {
        'CategoryName': 'Beverages',
        'ProductName': 'Chang',
        'QuantityPerUnit': '24 - 12 oz bottles',
        'UnitsInStock': 17,
        'Discontinued': true
    },
    {

```

```

    'CategoryName': 'Beverages',
    'ProductName': 'Chartreuse verte',
    'QuantityPerUnit': '750 cc per bottle',
    'UnitsInStock': 69,
    'Discontinued': true
  },
  {
    'CategoryName': 'Beverages',
    'ProductName': 'C\u00f4te de Blaye',
    'QuantityPerUnit': '12 - 75 cl bottles',
    'UnitsInStock': 17,
    'Discontinued': false
  },
  {
    'CategoryName': 'Beverages',
    'ProductName': 'Ipoh Coffee',
    'QuantityPerUnit': '16 - 500 g tins',
    'UnitsInStock': 17,
    'Discontinued': true
  },
  {
    'CategoryName': 'Beverages',
    'ProductName': 'Lakkalik\u00f6\u00f6ri',
    'QuantityPerUnit': '500 ml',
    'UnitsInStock': 57,
    'Discontinued': true
  },
  {
    'CategoryName': 'Beverages',
    'ProductName': 'Laughing Lumberjack Lager',
    'QuantityPerUnit': '24 - 12 oz bottles',
    'UnitsInStock': 52,
    'Discontinued': false
  },
  {
    'CategoryName': 'Beverages',
    'ProductName': 'Outback Lager',
    'QuantityPerUnit': '24 - 355 ml bottles',
    'UnitsInStock': 15,
    'Discontinued': false
  },
  {
    'CategoryName': 'Beverages',
    'ProductName': 'Rh\u00f6nbr\u00e4u Klosterbier',
    'QuantityPerUnit': '24 - 0.5 l bottles',
    'UnitsInStock': 125,
    'Discontinued': false
  },
  {
    'CategoryName': 'Beverages',
    'ProductName': 'Sasquatch Ale',
    'QuantityPerUnit': '24 - 12 oz bottles',
    'UnitsInStock': 111,
    'Discontinued': true
  },
  {
    'CategoryName': 'Beverages',

```

```

    'ProductName': 'Steeleye Stout',
    'QuantityPerUnit': '24 - 12 oz bottles',
    'UnitsInStock': 20,
    'Discontinued': true
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Aniseed Syrup',
    'QuantityPerUnit': '12 - 550 ml bottles',
    'UnitsInStock': 13,
    'Discontinued': true
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Chef Anton\'s Cajun Seasoning',
    'QuantityPerUnit': '48 - 6 oz jars',
    'UnitsInStock': 53,
    'Discontinued': false
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Genen Shouyu',
    'QuantityPerUnit': '24 - 250 ml bottles',
    'UnitsInStock': 39,
    'Discontinued': false
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Grandma\'s Boysenberry Spread',
    'QuantityPerUnit': '12 - 8 oz jars',
    'UnitsInStock': 120,
    'Discontinued': false
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Gula Malacca',
    'QuantityPerUnit': '20 - 2 kg bags',
    'UnitsInStock': 27,
    'Discontinued': true
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Louisiana Fiery Hot Pepper Sauce',
    'QuantityPerUnit': '32 - 8 oz bottles',
    'UnitsInStock': 76,
    'Discontinued': false
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Louisiana Hot Spiced Okra',
    'QuantityPerUnit': '24 - 8 oz jars',
    'UnitsInStock': 4,
    'Discontinued': false
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Northwoods Cranberry Sauce',

```

```

    'QuantityPerUnit': '12 - 12 oz jars',
    'UnitsInStock': 6,
    'Discontinued': true
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Original Frankfurter gr\u00f6\u00dfe',
    'QuantityPerUnit': '12 boxes',
    'UnitsInStock': 32,
    'Discontinued': false
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Sirop d\u00e9rable',
    'QuantityPerUnit': '24 - 500 ml bottles',
    'UnitsInStock': 113,
    'Discontinued': false
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Vegie-spread',
    'QuantityPerUnit': '15 - 625 g jars',
    'UnitsInStock': 24,
    'Discontinued': false
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'Chocolade',
    'QuantityPerUnit': '10 pkgs.',
    'UnitsInStock': 15,
    'Discontinued': true
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'Gumb\u00e4r Gummib\u00e4rchen',
    'QuantityPerUnit': '100 - 250 g bags',
    'UnitsInStock': 15,
    'Discontinued': false
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'Maxilaku',
    'QuantityPerUnit': '24 - 50 g pkgs.',
    'UnitsInStock': 10,
    'Discontinued': false
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'NuNuCa Nu\u00df-Nougat-Creme',
    'QuantityPerUnit': '20 - 450 g glasses',
    'UnitsInStock': 76,
    'Discontinued': true
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'Pavlova',
    'QuantityPerUnit': '32 - 500 g boxes',

```

```

    'UnitsInStock': 29,
    'Discontinued': true
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'Schoggi Schokolade',
    'QuantityPerUnit': '100 - 100 g pieces',
    'UnitsInStock': 49,
    'Discontinued': true
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'Scottish Longbreads',
    'QuantityPerUnit': '10 boxes x 8 pieces',
    'UnitsInStock': 6,
    'Discontinued': true
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'Sir Rodney\'s Marmalade',
    'QuantityPerUnit': '30 gift boxes',
    'UnitsInStock': 40,
    'Discontinued': true
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'Sir Rodney\'s Scones',
    'QuantityPerUnit': '24 pkgs. x 4 pieces',
    'UnitsInStock': 3,
    'Discontinued': true
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'Tarte au sucre',
    'QuantityPerUnit': '48 pies',
    'UnitsInStock': 17,
    'Discontinued': false
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'Teatime Chocolate Biscuits',
    'QuantityPerUnit': '10 boxes x 12 pieces',
    'UnitsInStock': 25,
    'Discontinued': false
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'Valkoinen suklaa',
    'QuantityPerUnit': '12 - 100 g bars',
    'UnitsInStock': 65,
    'Discontinued': false
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'Zaanse koeken',
    'QuantityPerUnit': '10 - 4 oz boxes',
    'UnitsInStock': 36,

```



```

    'Discontinued': true
  },
  {
    'CategoryName': 'Dairy Products',
    'ProductName': 'Camembert Pierrot',
    'QuantityPerUnit': '15 - 300 g rounds',
    'UnitsInStock': 19,
    'Discontinued': false
  },
  {
    'CategoryName': 'Dairy Products',
    'ProductName': 'Flotemysost',
    'QuantityPerUnit': '10 - 500 g pkgs.',
    'UnitsInStock': 26,
    'Discontinued': false
  },
  {
    'CategoryName': 'Dairy Products',
    'ProductName': 'Geitost',
    'QuantityPerUnit': '500 g',
    'UnitsInStock': 112,
    'Discontinued': false
  },
  {
    'CategoryName': 'Dairy Products',
    'ProductName': 'Gorgonzola Telino',
    'QuantityPerUnit': '12 - 100 g pkgs',
    'UnitsInStock': 0,
    'Discontinued': true
  },
  {
    'CategoryName': 'Dairy Products',
    'ProductName': 'Gudbrandsdalsost',
    'QuantityPerUnit': '10 kg pkg.',
    'UnitsInStock': 26,
    'Discontinued': true
  },
  {
    'CategoryName': 'Dairy Products',
    'ProductName': 'Mascarpone Fabioli',
    'QuantityPerUnit': '24 - 200 g pkgs.',
    'UnitsInStock': 9,
    'Discontinued': true
  },
  {
    'CategoryName': 'Dairy Products',
    'ProductName': 'Mozzarella di Giovanni',
    'QuantityPerUnit': '24 - 200 g pkgs.',
    'UnitsInStock': 14,
    'Discontinued': true
  },
  {
    'CategoryName': 'Dairy Products',
    'ProductName': 'Queso Cabrales',
    'QuantityPerUnit': '1 kg pkg.',
    'UnitsInStock': 22,
    'Discontinued': true
  }

```

```

},
{
  'CategoryName': 'Dairy Products',
  'ProductName': 'Queso Manchego La Pastora',
  'QuantityPerUnit': '10 - 500 g pkgs.',
  'UnitsInStock': 86,
  'Discontinued': true
},
{
  'CategoryName': 'Dairy Products',
  'ProductName': 'Raclette Courdavault',
  'QuantityPerUnit': '5 kg pkg.',
  'UnitsInStock': 79,
  'Discontinued': false
},
{
  'CategoryName': 'Grains/Cereals',
  'ProductName': 'Filo Mix',
  'QuantityPerUnit': '16 - 2 kg boxes',
  'UnitsInStock': 38,
  'Discontinued': false
},
{
  'CategoryName': 'Grains/Cereals',
  'ProductName': 'Gnocchi di nonna Alice',
  'QuantityPerUnit': '24 - 250 g pkgs.',
  'UnitsInStock': 21,
  'Discontinued': false
},
{
  'CategoryName': 'Grains/Cereals',
  'ProductName': 'Gustaf\'s Kn\u00e4ckebr\u00f6d',
  'QuantityPerUnit': '24 - 500 g pkgs.',
  'UnitsInStock': 104,
  'Discontinued': true
},
{
  'CategoryName': 'Grains/Cereals',
  'ProductName': 'Ravioli Angelo',
  'QuantityPerUnit': '24 - 250 g pkgs.',
  'UnitsInStock': 36,
  'Discontinued': true
},
{
  'CategoryName': 'Grains/Cereals',
  'ProductName': 'Tunnbr\u00f6d',
  'QuantityPerUnit': '12 - 250 g pkgs.',
  'UnitsInStock': 61,
  'Discontinued': true
},
{
  'CategoryName': 'Grains/Cereals',
  'ProductName': 'Wimmers gute Semmelkn\u00f6del',
  'QuantityPerUnit': '20 bags x 4 pieces',
  'UnitsInStock': 22,
  'Discontinued': true
},
},

```

```

{
  'CategoryName': 'Meat/Poultry',
  'ProductName': 'P\u00e2t\u00e9 chinois',
  'QuantityPerUnit': '24 boxes x 2 pies',
  'UnitsInStock': 115,
  'Discontinued': false
},
{
  'CategoryName': 'Meat/Poultry',
  'ProductName': 'Tourti\u00e8re',
  'QuantityPerUnit': '16 pies',
  'UnitsInStock': 21,
  'Discontinued': false
},
{
  'CategoryName': 'Produce',
  'ProductName': 'Longlife Tofu',
  'QuantityPerUnit': '5 kg pkg.',
  'UnitsInStock': 4,
  'Discontinued': false
},
{
  'CategoryName': 'Produce',
  'ProductName': 'Manjimup Dried Apples',
  'QuantityPerUnit': '50 - 300 g pkgs.',
  'UnitsInStock': 20,
  'Discontinued': false
},
{
  'CategoryName': 'Produce',
  'ProductName': 'Tofu',
  'QuantityPerUnit': '40 - 100 g pkgs.',
  'UnitsInStock': 35,
  'Discontinued': true
},
{
  'CategoryName': 'Produce',
  'ProductName': 'Uncle Bob\'s Organic Dried Pears',
  'QuantityPerUnit': '12 - 1 lb pkgs.',
  'UnitsInStock': 15,
  'Discontinued': true
},
{
  'CategoryName': 'Seafood',
  'ProductName': 'Boston Crab Meat',
  'QuantityPerUnit': '24 - 4 oz tins',
  'UnitsInStock': 123,
  'Discontinued': true
},
{
  'CategoryName': 'Seafood',
  'ProductName': 'Carnarvon Tigers',
  'QuantityPerUnit': '16 kg pkg.',
  'UnitsInStock': 42,
  'Discontinued': true
},
{

```

```

    'CategoryName': 'Seafood',
    'ProductName': 'Escargots de Bourgogne',
    'QuantityPerUnit': '24 pieces',
    'UnitsInStock': 62,
    'Discontinued': true
  },
  {
    'CategoryName': 'Seafood',
    'ProductName': 'Gravad lax',
    'QuantityPerUnit': '12 - 500 g pkgs.',
    'UnitsInStock': 11,
    'Discontinued': true
  },
  {
    'CategoryName': 'Seafood',
    'ProductName': 'Ikura',
    'QuantityPerUnit': '12 - 200 ml jars',
    'UnitsInStock': 31,
    'Discontinued': false
  },
  {
    'CategoryName': 'Seafood',
    'ProductName': 'Inlagd Sill',
    'QuantityPerUnit': '24 - 250 g jars',
    'UnitsInStock': 112,
    'Discontinued': false
  },
  {
    'CategoryName': 'Seafood',
    'ProductName': 'Jack\'s New England Clam Chowder',
    'QuantityPerUnit': '12 - 12 oz cans',
    'UnitsInStock': 85,
    'Discontinued': false
  },
  {
    'CategoryName': 'Seafood',
    'ProductName': 'Konbu',
    'QuantityPerUnit': '2 kg box',
    'UnitsInStock': 24,
    'Discontinued': false
  },
  {
    'CategoryName': 'Seafood',
    'ProductName': 'Nord-Ost Matjeshering',
    'QuantityPerUnit': '10 - 200 g glasses',
    'UnitsInStock': 10,
    'Discontinued': false
  },
  {
    'CategoryName': 'Seafood',
    'ProductName': 'R\u00f6d Kaviar',
    'QuantityPerUnit': '24 - 150 g jars',
    'UnitsInStock': 101,
    'Discontinued': false
  },
  {
    'CategoryName': 'Seafood',

```

```

        'ProductName': 'Rogede sild',
        'QuantityPerUnit': '1k pkg.',
        'UnitsInStock': 5,
        'Discontinued': false
    },
    {
        'CategoryName': 'Seafood',
        'ProductName': 'Spegesild',
        'QuantityPerUnit': '4 - 450 g glasses',
        'UnitsInStock': 95,
        'Discontinued': true
    }
];
export const customerData = [
    {
        'CustomerID': 'ALFKI',
        'ContactName': 'Maria ',
        'CompanyName': 'Alfreds Futterkiste',
        'Address': 'Obere Str. 57',
        'Country': 'Germany'
    },
    {
        'CustomerID': 'ANATR',
        'ContactName': 'Ana Trujillo',
        'CompanyName': 'Ana Trujillo Emparedados y helados',
        'Address': 'Avda. de la Constitución 2222',
        'Country': 'Mexico'
    },
    {
        'CustomerID': 'ANTON',
        'ContactName': 'Antonio Moreno',
        'CompanyName': 'Antonio Moreno Taquería',
        'Address': 'Mataderos 2312',
        'Country': 'Mexico'
    },
    {
        'CustomerID': 'AROUT',
        'ContactName': 'Thomas Hardy',
        'CompanyName': 'Around the Horn',
        'Address': '120 Hanover Sq.',
        'Country': 'UK'
    },
    {
        'CustomerID': 'BERGS',
        'ContactName': 'Christina Berglund',
        'CompanyName': 'Berglunds snabbköp',
        'Address': 'Berguvsvägen 8',
        'Country': 'Sweden'
    },
    {
        'CustomerID': 'BLAUS',
        'ContactName': 'Hanna Moos',
        'CompanyName': 'Blauer See Delikatessen',
        'Address': 'Forsterstr. 57',
        'Country': 'Germany'
    },
    {

```

```

    'CustomerID': 'BLONP',
    'ContactName': 'Frédérique Citeaux',
    'CompanyName': 'Blondesddsl père et fils',
    'Address': '24, place Kléber',
    'Country': 'France'
  },
  {
    'CustomerID': 'BOLID',
    'ContactName': 'Martín Sommer',
    'CompanyName': 'Bólido Comidas preparadas',
    'Address': 'C/ Araquil, 67',
    'Country': 'Spain'
  },
  {
    'CustomerID': 'BONAP',
    'ContactName': 'Laurence Lebihan',
    'CompanyName': 'Bon app',
    'Address': '12, rue des Bouchers',
    'Country': 'France'
  },
  {
    'CustomerID': 'BOTTM',
    'ContactName': 'Elizabeth Lincoln',
    'CompanyName': 'Bottom-Dollar Markets',
    'Address': '23 Tsawassen Blvd.',
    'Country': 'Canada'
  },
  {
    'CustomerID': 'BSBEV',
    'ContactName': 'Victoria Ashworth',
    'CompanyName': 'B\'s Beverages',
    'Address': 'Fauntleroy Circus',
    'Country': 'UK'
  },
  {
    'CustomerID': 'CACTU',
    'ContactName': 'Patricio Simpson',
    'CompanyName': 'Cactus Comidas para llevar',
    'Address': 'Cerrito 333',
    'Country': 'Argentina'
  },
  {
    'CustomerID': 'CENTC',
    'ContactName': 'Francisco Chang',
    'CompanyName': 'Centro comercial Moctezuma',
    'Address': 'Sierras de Granada 9993',
    'Country': 'Mexico'
  },
  {
    'CustomerID': 'CHOPS',
    'ContactName': 'Yang Wang',
    'CompanyName': 'Chop-suey Chinese',
    'Address': 'Hauptstr. 29',
    'Country': 'Switzerland'
  },
  {
    'CustomerID': 'COMMI',

```

```

    'ContactName': 'Pedro Afonso',
    'CompanyName': 'Comércio Mineiro',
    'Address': 'Av. dos Lusíadas, 23',
    'Country': 'Brazil'
  },
  {
    'CustomerID': 'CONSH',
    'ContactName': 'Elizabeth Brown',
    'CompanyName': 'Consolidated Holdings',
    'Address': 'Berkeley Gardens 12 Brewery',
    'Country': 'UK'
  },
  {
    'CustomerID': 'DRACD',
    'ContactName': 'Sven Ottlieb',
    'CompanyName': 'Drachenblut Delikatessen',
    'Address': 'Walserweg 21',
    'Country': 'Germany'
  },
  {
    'CustomerID': 'DUMON',
    'ContactName': 'Janine Labrune',
    'CompanyName': 'Du monde entier',
    'Address': '67, rue des Cinquante Otages',
    'Country': 'France'
  },
  {
    'CustomerID': 'EASTC',
    'ContactName': 'Ann Devon',
    'CompanyName': 'Eastern Connection',
    'Address': '35 King George',
    'Country': 'UK'
  },
  {
    'CustomerID': 'ERNSH',
    'ContactName': 'Roland Mendel',
    'CompanyName': 'Ernst Handel',
    'Address': 'Kirchgasse 6',
    'Country': 'Austria'
  },
  {
    'CustomerID': 'FAMIA',
    'ContactName': 'Aria Cruz',
    'CompanyName': 'Familia Arquibaldo',
    'Address': 'Rua Orós, 92',
    'Country': 'Brazil'
  },
  {
    'CustomerID': 'FISSA',
    'ContactName': 'Diego Roel',
    'CompanyName': 'FISSA Fabrica Inter. Salchichas S.A.',
    'Address': 'C/ Moralzarzal, 86',
    'Country': 'Spain'
  },
  {
    'CustomerID': 'FOLIG',
    'ContactName': 'Martine Rancé',

```

```

    'CompanyName': 'Folies gourmandes',
    'Address': '184, chaussée de Tournai',
    'Country': 'France'
  },
  {
    'CustomerID': 'FOLKO',
    'ContactName': 'Maria Larsson',
    'CompanyName': 'Folk och fä HB',
    'Address': 'Åkergatan 24',
    'Country': 'Sweden'
  },
  {
    'CustomerID': 'FRANK',
    'ContactName': 'Peter Franken',
    'CompanyName': 'Frankenversand',
    'Address': 'Berliner Platz 43',
    'Country': 'Germany'
  },
  {
    'CustomerID': 'FRANR',
    'ContactName': 'Carine Schmitt',
    'CompanyName': 'France restauration',
    'Address': '54, rue Royale',
    'Country': 'France'
  },
  {
    'CustomerID': 'FRANS',
    'ContactName': 'Paolo Accorti',
    'CompanyName': 'Franchi S.p.A.',
    'Address': 'Via Monte Bianco 34',
    'Country': 'Italy'
  },
  {
    'CustomerID': 'FURIB',
    'ContactName': 'Lino Rodriguez',
    'CompanyName': 'Furia Bacalhau e Frutos do Mar',
    'Address': 'Jardim das rosas n. 32',
    'Country': 'Portugal'
  },
  {
    'CustomerID': 'GALED',
    'ContactName': 'Eduardo Saavedra',
    'CompanyName': 'Galería del gastrónomo',
    'Address': 'Rambla de Cataluña, 23',
    'Country': 'Spain'
  },
  {
    'CustomerID': 'GODOS',
    'ContactName': 'José Pedro Freyre',
    'CompanyName': 'Godos Cocina Típica',
    'Address': 'C/ Romero, 33',
    'Country': 'Spain'
  },
  {
    'CustomerID': 'GOURL',
    'ContactName': 'André Fonseca',
    'CompanyName': 'Gourmet Lanchonetes',

```



```

    'Address': 'Av. Brasil, 442',
    'Country': 'Brazil'
  },
  {
    'CustomerID': 'GREAL',
    'ContactName': 'Howard Snyder',
    'CompanyName': 'Great Lakes Food Market',
    'Address': '2732 Baker Blvd.',
    'Country': 'USA'
  },
  {
    'CustomerID': 'GROSR',
    'ContactName': 'Manuel Pereira',
    'CompanyName': 'GROSELLA-Restaurante',
    'Address': '5ª Ave. Los Palos Grandes',
    'Country': 'Venezuela'
  },
  {
    'CustomerID': 'HANAR',
    'ContactName': 'Mario Pontes',
    'CompanyName': 'Hanari Carnes',
    'Address': 'Rua do Paço, 67',
    'Country': 'Brazil'
  },
  {
    'CustomerID': 'HILAA',
    'ContactName': 'Carlos Hernández',
    'CompanyName': 'HILARION-Abastos',
    'Address': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'Country': 'Venezuela'
  },
  {
    'CustomerID': 'HUNGC',
    'ContactName': 'Yoshi Latimer',
    'CompanyName': 'Hungry Coyote Import Store',
    'Address': 'City Center Plaza 516 Main St.',
    'Country': 'USA'
  },
  {
    'CustomerID': 'HUNGO',
    'ContactName': 'Patricia McKenna',
    'CompanyName': 'Hungry Owl All-Night Grocers',
    'Address': '8 Johnstown Road',
    'Country': 'Ireland'
  },
  {
    'CustomerID': 'ISLAT',
    'ContactName': 'Helen Bennett',
    'CompanyName': 'Island Trading',
    'Address': 'Garden House Crowther Way',
    'Country': 'UK'
  },
  {
    'CustomerID': 'KOENE',
    'ContactName': 'Philip Cramer',
    'CompanyName': 'Königlich Essen',
    'Address': 'Maubelstr. 90',

```

```

    'Country': 'Germany'
  },
  {
    'CustomerID': 'LACOR',
    'ContactName': 'Daniel Tonini',
    'CompanyName': 'La corne d\'abondance',
    'Address': '67, avenue de l\'Europe',
    'Country': 'France'
  },
  {
    'CustomerID': 'LAMAI',
    'ContactName': 'Annette Roulet',
    'CompanyName': 'La maison d\'Asie',
    'Address': '1 rue Alsace-Lorraine',
    'Country': 'France'
  },
  {
    'CustomerID': 'LAUGB',
    'ContactName': 'Yoshi Tannamuri',
    'CompanyName': 'Laughing Bacchus Wine Cellars',
    'Address': '1900 Oak St.',
    'Country': 'Canada'
  },
  {
    'CustomerID': 'LAZYK',
    'ContactName': 'John Steel',
    'CompanyName': 'Lazy K Kountry Store',
    'Address': '12 Orchestra Terrace',
    'Country': 'USA'
  },
  {
    'CustomerID': 'LEHMS',
    'ContactName': 'Renate Messner',
    'CompanyName': 'Lehmanns Marktstand',
    'Address': 'Magazinweg 7',
    'Country': 'Germany'
  },
  {
    'CustomerID': 'LETSS',
    'ContactName': 'Jaime Yorres',
    'CompanyName': 'Let\'s Stop N Shop',
    'Address': '87 Polk St. Suite 5',
    'Country': 'USA'
  },
  {
    'CustomerID': 'LILAS',
    'ContactName': 'Carlos González',
    'CompanyName': 'LILA-Supermercado',
    'Address': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'Country': 'Venezuela'
  },
  {
    'CustomerID': 'LINOD',
    'ContactName': 'Felipe Izquierdo',
    'CompanyName': 'LINO-Delicateses',
    'Address': 'Ave. 5 de Mayo Porlamar',
    'Country': 'Venezuela'
  }

```

```

    },
    {
        'CustomerID': 'LONEP',
        'ContactName': 'Fran Wilson',
        'CompanyName': 'Lonesome Pine Restaurant',
        'Address': '89 Chiaroscuro Rd.',
        'Country': 'USA'
    },
    {
        'CustomerID': 'MAGAA',
        'ContactName': 'Giovanni Rovelli',
        'CompanyName': 'Magazzini Alimentari Riuniti',
        'Address': 'Via Ludovico il Moro 22',
        'Country': 'Italy'
    },
    {
        'CustomerID': 'MAISD',
        'ContactName': 'Catherine Dewey',
        'CompanyName': 'Maison Dewey',
        'Address': 'Rue Joseph-Bens 532',
        'Country': 'Belgium'
    },
    {
        'CustomerID': 'MEREP',
        'ContactName': 'Jean Fresnière',
        'CompanyName': 'Mère Paillarde',
        'Address': '43 rue St. Laurent',
        'Country': 'Canada'
    },
    {
        'CustomerID': 'MORGK',
        'ContactName': 'Alexander Feuer',
        'CompanyName': 'Morgenstern Gesundkost',
        'Address': 'Heerstr. 22',
        'Country': 'Germany'
    },
    {
        'CustomerID': 'NORTS',
        'ContactName': 'Simon Crowther',
        'CompanyName': 'North/South',
        'Address': 'South House 300 Queensbridge',
        'Country': 'UK'
    },
    {
        'CustomerID': 'OCEAN',
        'ContactName': 'Yvonne Moncada',
        'CompanyName': 'Océano Atlántico Ltda.',
        'Address': 'Ing. Gustavo Moncada 8585 Piso 20-A',
        'Country': 'Argentina'
    },
    {
        'CustomerID': 'OLDWO',
        'ContactName': 'Rene Phillips',
        'CompanyName': 'Old World Delicatessen',
        'Address': '2743 Bering St.',
        'Country': 'USA'
    },
    },

```

```

{
    'CustomerID': 'OTTIK',
    'ContactName': 'Henriette Pfalzheim',
    'CompanyName': 'Ottilies Käseladen',
    'Address': 'Mehrheimerstr. 369',
    'Country': 'Germany'
},
{
    'CustomerID': 'PARIS',
    'ContactName': 'Marie Bertrand',
    'CompanyName': 'Paris spécialités',
    'Address': '265, boulevard Charonne',
    'Country': 'France'
},
{
    'CustomerID': 'PERIC',
    'ContactName': 'Guillermo Fernández',
    'CompanyName': 'Pericles Comidas clásicas',
    'Address': 'Calle Dr. Jorge Cash 321',
    'Country': 'Mexico'
},
{
    'CustomerID': 'PICCO',
    'ContactName': 'Georg Pipps',
    'CompanyName': 'Piccolo und mehr',
    'Address': 'Geislweg 14',
    'Country': 'Austria'
},
{
    'CustomerID': 'PRINI',
    'ContactName': 'Isabel de Castro',
    'CompanyName': 'Princesa Isabel Vinhos',
    'Address': 'Estrada da saúde n. 58',
    'Country': 'Portugal'
},
{
    'CustomerID': 'QUEDE',
    'ContactName': 'Bernardo Batista',
    'CompanyName': 'Que Delícia',
    'Address': 'Rua da Panificadora, 12',
    'Country': 'Brazil'
},
{
    'CustomerID': 'QUEEN',
    'ContactName': 'Lúcia Carvalho',
    'CompanyName': 'Queen Cozinha',
    'Address': 'Alameda dos Canários, 891',
    'Country': 'Brazil'
},
{
    'CustomerID': 'QUICK',
    'ContactName': 'Horst Kloss',
    'CompanyName': 'QUICK-Stop',
    'Address': 'Taucherstraße 10',
    'Country': 'Germany'
},
{

```

```

    'CustomerID': 'RANCH',
    'ContactName': 'Sergio Gutiérrez',
    'CompanyName': 'Rancho grande',
    'Address': 'Av. del Libertador 900',
    'Country': 'Argentina'
  },
  {
    'CustomerID': 'RATTC',
    'ContactName': 'Paula Wilson',
    'CompanyName': 'Rattlesnake Canyon Grocery',
    'Address': '2817 Milton Dr.',
    'Country': 'USA'
  },
  {
    'CustomerID': 'REGGC',
    'ContactName': 'Maurizio Moroni',
    'CompanyName': 'Reggiani Caseifici',
    'Address': 'Strada Provinciale 124',
    'Country': 'Italy'
  },
  {
    'CustomerID': 'RICAR',
    'ContactName': 'Janete Limeira',
    'CompanyName': 'Ricardo Adocicados',
    'Address': 'Av. Copacabana, 267',
    'Country': 'Brazil'
  },
  {
    'CustomerID': 'RICSU',
    'ContactName': 'Michael Holz',
    'CompanyName': 'Richter Supermarkt',
    'Address': 'Grenzacherweg 237',
    'Country': 'Switzerland'
  },
  {
    'CustomerID': 'ROMEY',
    'ContactName': 'Alejandra Camino',
    'CompanyName': 'Romero y tomillo',
    'Address': 'Gran Vía, 1',
    'Country': 'Spain'
  },
  {
    'CustomerID': 'SANTG',
    'ContactName': 'Jonas Bergulfsen',
    'CompanyName': 'Santé Gourmet',
    'Address': 'Erling Skakkes gate 78',
    'Country': 'Norway'
  },
  {
    'CustomerID': 'SAVEA',
    'ContactName': 'Jose Pavarotti',
    'CompanyName': 'Save-a-lot Markets',
    'Address': '187 Suffolk Ln.',
    'Country': 'USA'
  },
  {
    'CustomerID': 'SEVES',

```

```

    'ContactName': 'Hari Kumar',
    'CompanyName': 'Seven Seas Imports',
    'Address': '90 Wadhurst Rd.',
    'Country': 'UK'
  },
  {
    'CustomerID': 'SIMOB',
    'ContactName': 'Jytte Petersen',
    'CompanyName': 'Simons bistro',
    'Address': 'Vinbæltet 34',
    'Country': 'Denmark'
  },
  {
    'CustomerID': 'SPECB',
    'ContactName': 'Dominique Perrier',
    'CompanyName': 'Spécialités du monde',
    'Address': '25, rue Lauriston',
    'Country': 'France'
  },
  {
    'CustomerID': 'SPLIR',
    'ContactName': 'Art Braunschweiger',
    'CompanyName': 'Split Rail Beer & Ale',
    'Address': 'P.O. Box 555',
    'Country': 'USA'
  },
  {
    'CustomerID': 'SUPRD',
    'ContactName': 'Pascale Cartrain',
    'CompanyName': 'Suprêmes délices',
    'Address': 'Boulevard Tirou, 255',
    'Country': 'Belgium'
  },
  {
    'CustomerID': 'THEBI',
    'ContactName': 'Liz Nixon',
    'CompanyName': 'The Big Cheese',
    'Address': '89 Jefferson Way Suite 2',
    'Country': 'USA'
  },
  {
    'CustomerID': 'THECR',
    'ContactName': 'Liu Wong',
    'CompanyName': 'The Cracker Box',
    'Address': '55 Grizzly Peak Rd.',
    'Country': 'USA'
  },
  {
    'CustomerID': 'TOMSP',
    'ContactName': 'Karin Josephs',
    'CompanyName': 'Toms Spezialitäten',
    'Address': 'Luisenstr. 48',
    'Country': 'Germany'
  },
  {
    'CustomerID': 'TORTU',
    'ContactName': 'Miguel Angel Paolino',

```

```

    'CompanyName': 'Tortuga Restaurante',
    'Address': 'Avda. Azteca 123',
    'Country': 'Mexico'
  },
  {
    'CustomerID': 'TRADH',
    'ContactName': 'Anabela Domingues',
    'CompanyName': 'Tradição Hipermercados',
    'Address': 'Av. Inês de Castro, 414',
    'Country': 'Brazil'
  },
  {
    'CustomerID': 'TRAIH',
    'ContactName': 'Helvetius Nagy',
    'CompanyName': 'Trail\'s Head Gourmet Provisioners',
    'Address': '722 DaVinci Blvd.',
    'Country': 'USA'
  },
  {
    'CustomerID': 'VAFFE',
    'ContactName': 'Palle Ibsen',
    'CompanyName': 'Vaffeljernet',
    'Address': 'Smagsloget 45',
    'Country': 'Denmark'
  },
  {
    'CustomerID': 'VICTE',
    'ContactName': 'Mary Saveley',
    'CompanyName': 'Victuailles en stock',
    'Address': '2, rue du Commerce',
    'Country': 'France'
  },
  {
    'CustomerID': 'VINET',
    'ContactName': 'Paul Henriot',
    'CompanyName': 'Vins et alcools Chevalier',
    'Address': '59 rue de l\'Abbaye',
    'Country': 'France'
  },
  {
    'CustomerID': 'WANDK',
    'ContactName': 'Rita Müller',
    'CompanyName': 'Die Wandernde Kuh',
    'Address': 'Adenauerallee 900',
    'Country': 'Germany'
  },
  {
    'CustomerID': 'WARTH',
    'ContactName': 'Pirkko Koskitalo',
    'CompanyName': 'Wartian Herkku',
    'Address': 'Torikatu 38',
    'Country': 'Finland'
  },
  {
    'CustomerID': 'WELLI',
    'ContactName': 'Paula Parente',
    'CompanyName': 'Wellington Importadora',

```

```

        'Address': 'Rua do Mercado, 12',
        'Country': 'Brazil'
    },
    {
        'CustomerID': 'WHITC',
        'ContactName': 'Karl Jablonski',
        'CompanyName': 'White Clover Markets',
        'Address': '305 - 14th Ave. S. Suite 3B',
        'Country': 'USA'
    },
    {
        'CustomerID': 'WILMK',
        'ContactName': 'Matti Karttunen',
        'CompanyName': 'Wilman Kala',
        'Address': 'Keskuskatu 45',
        'Country': 'Finland'
    },
    {
        'CustomerID': 'WOLZA',
        'ContactName': 'Zbyszek Piestrzeniewicz',
        'CompanyName': 'Wolski Zajazd',
        'Address': 'ul. Filtrowa 68',
        'Country': 'Poland'
    }
];
export const data = orderData.map((item) => {
    let name = (<cType />)[ ] > customerData;
}).filter((cItem) => {
    return cItem.CustomerID === item.CustomerID;
})[0];
item.CustomerName = (name || <cType>).ContactName;
return item;
});
export const inventoryData: Object[] = [
    {
        'Inventor': 'Kia Silverbrook',
        'NumberofPatentFamilies': 4737,
        'Country': 'Australia',
        'Number of INPADOC patents': 9839,
        'Active': '1994-2016',
        'Mainfieldsofinvention': 'Printing, Digital paper, Internet, Electronics, Lab-on-a-chip, MEMS, Mechanical, VLSI',
    },
    {
        'Inventor': 'Shunpei Yamazaki',
        'NumberofPatentFamilies': 4677,
        'Country': 'Japan',
        'Number of INPADOC patents': '10000+',
        'Active': '1976-2016',
        'Mainfieldsofinvention': 'Thin film transistors, Liquid crystal displays, Solar cells, Flash memory, OLED',
    },
    {
        'Inventor': 'Lowell L. Wood, Jr.',
        'NumberofPatentFamilies': 1419,
        'Country': 'USA',
        'Number of INPADOC patents': 1332,
        'Active': '1977-2016',
        'Mainfieldsofinvention': 'Mosquito laser, Nuclear weapons',
    },
];

```



```

    {'Inventor': 'Paul Lapstun',
      'NumberofPatentFamilies': 1281,
      'Country': 'Australia',
      'Number of INPADOC patents': 3099,
      'Active': '2000-2016',
      'Mainfieldsofinvention': 'Printing, Digital paper, Internet,
Electronics, CGI, VLSI',
    },
    {'Inventor': 'Gurtej Sandhu',
      'NumberofPatentFamilies': 1255,
      'Country': 'India',
      'Number of INPADOC patents': 2038,
      'Active': '1991-2016',
      'Mainfieldsofinvention': 'Thin film processes and materials, VLSI,
Semiconductor device fabrication',
    },
    {'Inventor': 'Jun Koyama',
      'NumberofPatentFamilies': 1240,
      'Country': 'Japan',
      'Number of INPADOC patents': 4126,
      'Active': '1991-2016',
      'Mainfieldsofinvention': 'Thin film transistors, Liquid crystal
displays, OLED',
    },
    {'Inventor': 'Roderick A. Hyde',
      'NumberofPatentFamilies': 1240,
      'Country': 'USA',
      'Number of INPADOC patents': 3360,
      'Active': '2001-2016',
      'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'Leonard Forbes',
      'NumberofPatentFamilies': 1093,
      'Country': 'Canada',
      'Number of INPADOC patents': 1398,
      'Active': '1991-2016',
      'Mainfieldsofinvention': 'Semiconductor Memories, CCDs, Thin film
processes and materials, VLSI',
    },
    {'Inventor': 'Thomas Edison',
      'NumberofPatentFamilies': 1084,
      'Country': 'USA',
      'Number of INPADOC patents': 2332,
      'Active': '1847(b)-1931(d)',
      'Mainfieldsofinvention': 'Electric power, Lighting, Batteries,
Phonograph, Cement, Telegraphy, Mining',
    },
    {'Inventor': 'Donald E. Weder',
      'NumberofPatentFamilies': 999,
      'Country': 'USA',
      'Number of INPADOC patents': 1993,
      'Active': '1976-2015',
      'Mainfieldsofinvention': 'Florist supplies',
    },
    {'Inventor': 'George Albert Lyon',
      'NumberofPatentFamilies': 993,
      'Country': 'Canada',

```

```

    'Number of INPADOC patents': 'NA',
    'Active': '1882(b)-1961(d)',
    'Mainfieldsofinvention': 'Automotive, Stainless steel products',
  },
  {'Inventor': 'John F. O\'Connor',
    'NumberofPatentFamilies': 949,
    'Country': 'USA',
    'Number of INPADOC patents': 'NA',
    'Active': '1864(b)-1938(d)',
    'Mainfieldsofinvention': 'Railway draft gearing',
  },
  {'Inventor': 'Melvin De Groote',
    'NumberofPatentFamilies': 925,
    'Country': 'USA',
    'Number of INPADOC patents': 'NA',
    'Active': '1895(b)-1963(d)',
    'Mainfieldsofinvention': 'Chemical de-emulsifiers',
  },
  {'Inventor': 'Jay S. Walker',
    'NumberofPatentFamilies': 918,
    'Country': 'USA',
    'Number of INPADOC patents': 2206,
    'Active': '1998-2016',
    'Mainfieldsofinvention': 'Gaming machines',
  },
  {'Inventor': 'Edward K. Y. Jung',
    'NumberofPatentFamilies': 911,
    'Country': 'USA',
    'Number of INPADOC patents': 2254,
    'Active': '1996-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {'Inventor': 'Francis H. Richards',
    'NumberofPatentFamilies': 894,
    'Country': 'USA',
    'Number of INPADOC patents': 'NA',
    'Active': '1850(b)-19??(d)',
    'Mainfieldsofinvention': 'Mechanical, automation',
  },
  {'Inventor': 'Kangguo Cheng',
    'NumberofPatentFamilies': 884,
    'Country': 'USA',
    'Number of INPADOC patents': 1314,
    'Active': '2004-2016',
    'Mainfieldsofinvention': 'Semiconductor device fabrication,
Semiconductor memory, Semiconductor device',
  },
  {'Inventor': 'Clarence T. Tegreene',
    'NumberofPatentFamilies': 872,
    'Country': 'USA',
    'Number of INPADOC patents': 2255,
    'Active': '2000-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {'Inventor': 'Ahmadreza Rofougaran',
    'NumberofPatentFamilies': 808,
    'Country': 'USA',

```

```

    'Number of INPADOC patents': 1396,
    'Active': '2002-2016',
    'Mainfieldsofinvention': 'Radio Frequency Integrated Circuits',
  },
  {'Inventor': 'Shou-Shan Fan',
    'NumberofPatentFamilies': 805,
    'Country': 'China',
    'Number of INPADOC patents': 2120,
    'Active': '2006-2016',
    'Mainfieldsofinvention': 'Carbon nanotubes and applications of
carbon nanotubes',
  },
  {'Inventor': 'Michael J. Sullivan',
    'NumberofPatentFamilies': 788,
    'Country': 'USA',
    'Number of INPADOC patents': 1560,
    'Active': '1977-2016',
    'Mainfieldsofinvention': 'Golf balls',
  },
  {'Inventor': 'Rick Allen Hamilton II',
    'NumberofPatentFamilies': 773,
    'Country': 'USA',
    'Number of INPADOC patents': 1064,
    'Active': '1999-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {'Inventor': 'Warren Farnworth',
    'NumberofPatentFamilies': 770,
    'Country': 'USA',
    'Number of INPADOC patents': 931,
    'Active': '1990-2016',
    'Mainfieldsofinvention': 'Semiconductor packaging',
  },
  {'Inventor': 'Carleton Ellis',
    'NumberofPatentFamilies': 753,
    'Country': 'USA',
    'Number of INPADOC patents': 'NA',
    'Active': '1876(b)-1941(d)',
    'Mainfieldsofinvention': 'Margarine, Polyester, Anti-knock gasoline,
Paint stripper',
  },
  {'Inventor': 'William H. Eby',
    'NumberofPatentFamilies': 733,
    'Country': 'USA',
    'Number of INPADOC patents': 758,
    'Active': '1994-2016',
    'Mainfieldsofinvention': 'Transgenic soybeans',
  },
  {'Inventor': 'Hideo Ando',
    'NumberofPatentFamilies': 728,
    'Country': 'Japan',
    'Number of INPADOC patents': 2588,
    'Active': '1983-2016',
    'Mainfieldsofinvention': 'Optical recording',
  },
  {'Inventor': 'Salman Akram',
    'NumberofPatentFamilies': 728,

```

```

    'Country': 'USA',
    'Number of INPADOC patents': 915,
    'Active': '1995-2016',
    'Mainfieldsofinvention': 'Semiconductor packaging',
  },
  {'Inventor': 'George Spector',
    'NumberofPatentFamilies': 722,
    'Country': 'USA',
    'Number of INPADOC patents': 747,
    'Active': '1976-1998',
    'Mainfieldsofinvention': 'Gadgets, Toys',
  },
  {'Inventor': 'Jeyhan Karaoguz',
    'NumberofPatentFamilies': 721,
    'Country': 'USA',
    'Number of INPADOC patents': 1530,
    'Active': '1996-2016',
    'Mainfieldsofinvention': 'Wireless communications, Computer
networks',
  },
  {'Inventor': 'Elihu Thomson',
    'NumberofPatentFamilies': 696,
    'Country': 'UK',
    'Number of INPADOC patents': 'NA',
    'Active': '1853(b)-1937(d)',
    'Mainfieldsofinvention': 'Electric power, Arc lamp, Electric motors,
Lightning arrester, Arc welder',
  },
  {'Inventor': 'Austin L. Gurney',
    'NumberofPatentFamilies': 695,
    'Country': 'USA',
    'Number of INPADOC patents': 3909,
    'Active': '1999-2016',
    'Mainfieldsofinvention': 'Proteins, Antibodies',
  },
  {'Inventor': 'Tetsujiro Kondo',
    'NumberofPatentFamilies': 684,
    'Country': 'Japan',
    'Number of INPADOC patents': 4158,
    'Active': '1987-2015',
    'Mainfieldsofinvention': 'Signal processing, Image processing',
  },
  {'Inventor': 'Nathan Myhrvold',
    'NumberofPatentFamilies': 661,
    'Country': 'USA',
    'Number of INPADOC patents': 1690,
    'Active': '1994-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {'Inventor': 'William I. Wood',
    'NumberofPatentFamilies': 653,
    'Country': 'USA',
    'Number of INPADOC patents': 3560,
    'Active': '1981-2016',
    'Mainfieldsofinvention': 'Proteins, Antibodies',
  },
  {'Inventor': 'Simon R. Walmsley',

```

```

        'NumberofPatentFamilies': 651,
        'Country': 'Australia',
        'Number of INPADOC patents': 1249,
        'Active': '1995-2015',
        'Mainfieldsofinvention': 'Printing, Electronics, VLSI,
Cryptography',
    },
    {'Inventor': 'Mark Malamud',
        'NumberofPatentFamilies': 632,
        'Country': 'USA',
        'Number of INPADOC patents': 1759,
        'Active': '1997-2016',
        'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'Royce A. Levien',
        'NumberofPatentFamilies': 630,
        'Country': 'USA',
        'Number of INPADOC patents': 1799,
        'Active': '1997-2016',
        'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'Audrey D. Goddard',
        'NumberofPatentFamilies': 622,
        'Country': 'USA',
        'Number of INPADOC patents': 3416,
        'Active': '1997-2014',
        'Mainfieldsofinvention': 'Proteins, Antibodies',
    },
    {'Inventor': 'Muriel Y. Ishikawa',
        'NumberofPatentFamilies': 619,
        'Country': 'USA',
        'Number of INPADOC patents': 1660,
        'Active': '2002-2016',
        'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'Robert W. Lord',
        'NumberofPatentFamilies': 618,
        'Country': 'USA',
        'Number of INPADOC patents': 1708,
        'Active': '2003-2016',
        'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'Jerome Lemelson',
        'NumberofPatentFamilies': 606,
        'Country': 'USA',
        'Number of INPADOC patents': 'NA',
        'Active': '1923(b)-1997(d)',
        'Mainfieldsofinvention': 'Toys, Industrial robots, Cordless
telephones, Fax machines, Videocassette recorders',
    },
    {'Inventor': 'Béla Barényi',
        'NumberofPatentFamilies': 595,
        'Country': 'Austria',
        'Number of INPADOC patents': 1244,
        'Active': '1907(b)-1997(d)',
        'Mainfieldsofinvention': 'Passive safety in automobiles',
    },
    },

```

```

    {'Inventor': 'Kie Y Ahn',
      'NumberofPatentFamilies': 593,
      'Country': 'USA',
      'Number of INPADOC patents': 709,
      'Active': '1976-2016',
      'Mainfieldsofinvention': 'Thin film processes and materials, VLSI,
Semiconductor device fabrication',
    },
    {'Inventor': 'Tadahiro Ohmi',
      'NumberofPatentFamilies': 592,
      'Country': 'Japan',
      'Number of INPADOC patents': 2691,
      'Active': '1981-2016',
      'Mainfieldsofinvention': 'Thin film processes and materials,
Semiconductor device fabrication',
    },
    {'Inventor': 'Jordin T. Kare',
      'NumberofPatentFamilies': 585,
      'Country': 'USA',
      'Number of INPADOC patents': 1559,
      'Active': '1992-2016',
      'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'Paul J. Godowski',
      'NumberofPatentFamilies': 579,
      'Country': 'USA',
      'Number of INPADOC patents': 2605,
      'Active': '1994-2014',
      'Mainfieldsofinvention': 'Proteins, Antibodies',
    },
    {'Inventor': 'Artur Fischer',
      'NumberofPatentFamilies': 570,
      'Country': 'Germany',
      'Number of INPADOC patents': 3097,
      'Active': '1976-2002',
      'Mainfieldsofinvention': 'Fasteners, Construction toys',
    },
    {'Inventor': 'Edward J. Nowak',
      'NumberofPatentFamilies': 564,
      'Country': 'USA',
      'Number of INPADOC patents': 1145,
      'Active': '1979-2016',
      'Mainfieldsofinvention': 'Semiconductor device fabrication,
Semiconductor memory, Semiconductor device',
    },
    {'Inventor': 'Louis L. Hsu',
      'NumberofPatentFamilies': 551,
      'Country': 'USA',
      'Number of INPADOC patents': 914,
      'Active': '1988-2016',
      'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'Edwin H. Land',
      'NumberofPatentFamilies': 535,
      'Country': 'USA',
      'Number of INPADOC patents': 1236,
      'Active': '1909(b)-1991(d)',
    },

```

```

    'Mainfieldsofinvention': 'Instant photography, Polarizing film',
  },
  { 'Inventor': 'Henri Dreyfus',
    'NumberofPatentFamilies': 524,
    'Country': 'Switzerland',
    'Number of INPADOC patents': 2117,
    'Active': '1882(b)-1944(d)',
    'Mainfieldsofinvention': 'Polymers, Synthetic fibers, Dyes',
  },
  { 'Inventor': 'Bruce B. Doris',
    'NumberofPatentFamilies': 522,
    'Country': 'USA',
    'Number of INPADOC patents': 867,
    'Active': '1995-2016',
    'Mainfieldsofinvention': 'Integrated Circuits, CMOS, DRAM,
Semiconductor device fabrication',
  },
  { 'Inventor': 'Clyde C. Farmer',
    'NumberofPatentFamilies': 513,
    'Country': 'USA',
    'Number of INPADOC patents': 830,
    'Active': '18??(b)-19??(d)',
    'Mainfieldsofinvention': 'Railway air brakes',
  },
  { 'Inventor': 'Heinz Focke',
    'NumberofPatentFamilies': 512,
    'Country': 'Germany',
    'Number of INPADOC patents': 2896,
    'Active': '1976-2013',
    'Mainfieldsofinvention': 'Cigarette packaging',
  },
  { 'Inventor': 'Mark I. Gardner',
    'NumberofPatentFamilies': 511,
    'Country': 'USA',
    'Number of INPADOC patents': 587,
    'Active': '1994-2010',
    'Mainfieldsofinvention': 'Consumer electronics, Energy, Computers,
Semiconductors, Physics',
  },
  { 'Inventor': 'Ravi K. Arimilli',
    'NumberofPatentFamilies': 506,
    'Country': 'India',
    'Number of INPADOC patents': 767,
    'Active': '1992-2016',
    'Mainfieldsofinvention': 'Computer architecture, Semiconductor
memory, Cache coherence, Symmetric multiprocessing',
  },
  { 'Inventor': 'Louis H. Morin',
    'NumberofPatentFamilies': 503,
    'Country': 'USA',
    'Number of INPADOC patents': 720,
    'Active': '18??(b)-19??(d)',
    'Mainfieldsofinvention': 'Fasteners, Locks, Bobbins',
  },
  { 'Inventor': 'Tobin A. King',
    'NumberofPatentFamilies': 497,
    'Country': 'Australia',

```

```

    'Number of INPADOC patents': 1218,
    'Active': '2000-2015',
    'Mainfieldsofinvention': 'Printing, Digital paper, Mechanical',
  },
  {'Inventor': 'Eric C. Leuthardt',
    'NumberofPatentFamilies': 495,
    'Country': 'USA',
    'Number of INPADOC patents': 1274,
    'Active': '2006-2016',
    'Mainfieldsofinvention': 'Medical devices',
  },
  {'Inventor': 'Ali Khakifirooz',
    'NumberofPatentFamilies': 489,
    'Country': 'USA',
    'Number of INPADOC patents': 737,
    'Active': '2011-2016',
    'Mainfieldsofinvention': 'Integrated Circuits, CMOS, Semiconductor
device fabrication',
  },
  {'Inventor': 'Jack A. Mandelman',
    'NumberofPatentFamilies': 481,
    'Country': 'USA',
    'Number of INPADOC patents': 889,
    'Active': '1987-2014',
    'Mainfieldsofinvention': 'Various',
  },
  {'Inventor': 'Jeffrey P. Gambino',
    'NumberofPatentFamilies': 479,
    'Country': 'USA',
    'Number of INPADOC patents': 798,
    'Active': '1992-2016',
    'Mainfieldsofinvention': 'MEMS, CMOS, BiCMOS, DRAM, Image Sensors,
RF, Biosensors, 3D Integrated Circuits',
  },
  {'Inventor': 'John M. Santosuosso',
    'NumberofPatentFamilies': 473,
    'Country': 'USA',
    'Number of INPADOC patents': 683,
    'Active': '2001-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {'Inventor': 'James M. Hart',
    'NumberofPatentFamilies': 464,
    'Country': 'USA',
    'Number of INPADOC patents': 1145,
    'Active': '1988-2016',
    'Mainfieldsofinvention': 'Motor vehicle transmission',
  },
  {'Inventor': 'Eberhard Ammermann',
    'NumberofPatentFamilies': 451,
    'Country': 'Germany',
    'Number of INPADOC patents': 5178,
    'Active': '1979-2015',
    'Mainfieldsofinvention': 'Fungicides',
  },
  {'Inventor': 'Thomas E. Murray',
    'NumberofPatentFamilies': 449,

```



```

    'Country': 'USA',
    'Number of INPADOC patents': 462,
    'Active': '1860(b)-1929(d)',
    'Mainfieldsofinvention': 'Electrical, HVAC, Wheels, Metal working,
Light dimmer',
  },
  {'Inventor': 'Akira Nakazawa',
    'NumberofPatentFamilies': 445,
    'Country': 'Australia',
    'Number of INPADOC patents': 1340,
    'Active': '1980-2016',
    'Mainfieldsofinvention': 'Printing, Mechanical',
  },
  {'Inventor': 'Hongyong Zhang',
    'NumberofPatentFamilies': 440,
    'Country': 'Japan',
    'Number of INPADOC patents': 858,
    'Active': '1993-2016',
    'Mainfieldsofinvention': 'Thin film transistors, Liquid crystal
displays',
  },
  {'Inventor': 'Ronald S. Cok',
    'NumberofPatentFamilies': 436,
    'Country': 'USA',
    'Number of INPADOC patents': 747,
    'Active': '1986-2016',
    'Mainfieldsofinvention': 'OLED displays; image processing',
  },
  {'Inventor': 'fe',
    'NumberofPatentFamilies': 430,
    'Country': 'USA',
    'Number of INPADOC patents': 1759,
    'Active': '1983-2016',
    'Mainfieldsofinvention': 'Biotechnology, Drug delivery, Tissue
engineering',
  },
  {'Inventor': 'Scott H. Wittkopp',
    'NumberofPatentFamilies': 429,
    'Country': 'USA',
    'Number of INPADOC patents': 1010,
    'Active': '2001-2016',
    'Mainfieldsofinvention': 'Motor vehicle transmission',
  },
  {'Inventor': 'John Hays Hammond, Jr.',
    'NumberofPatentFamilies': 417,
    'Country': 'USA',
    'Number of INPADOC patents': 460,
    'Active': '1888(b)-1965(d)',
    'Mainfieldsofinvention': 'Radio control, Radio communications,
Torpedoes',
  },
  {'Inventor': 'Wilhelm Brandes',
    'NumberofPatentFamilies': 411,
    'Country': 'Germany',
    'Number of INPADOC patents': 2923,
    'Active': '1976-2010',
    'Mainfieldsofinvention': 'Fungicides',
  },

```

```

    },
    {'Inventor': 'Anthony K. Stamper',
     'NumberofPatentFamilies': 411,
     'Country': 'USA',
     'Number of INPADOC patents': 726,
     'Active': '1998-2016',
     'Mainfieldsofinvention': 'MEMS, CMOS, BiCMOS, Silicon-germanium',
    },
    {'Inventor': 'Hossein Eslambolchi',
     'NumberofPatentFamilies': 410,
     'Country': 'USA',
     'Number of INPADOC patents': 631,
     'Active': '1993-2016',
     'Mainfieldsofinvention': 'Telecommunications, Network intelligence,
information Technology, communications technology',
    },
    {'Inventor': 'Stanford R. Ovshinsky',
     'NumberofPatentFamilies': 400,
     'Country': 'USA',
     'Number of INPADOC patents': 1649,
     'Active': '1922(b)-2012(d)',
     'Mainfieldsofinvention': 'Batteries, Solar cells, Liquid crystal
displays, Hydrogen fuel cells, Computer data storage',
    },
    {'Inventor': 'Victoria Y. H. Wood',
     'NumberofPatentFamilies': 400,
     'Country': 'USA',
     'Number of INPADOC patents': 1045,
     'Active': '2009-2016',
     'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'Josef Theurer',
     'NumberofPatentFamilies': 388,
     'Country': 'Austria',
     'Number of INPADOC patents': 5085,
     'Active': '1976-2016',
     'Mainfieldsofinvention': 'Railroad maintenance machines',
    },
    {'Inventor': 'Cary L. Bates',
     'NumberofPatentFamilies': 384,
     'Country': 'USA',
     'Number of INPADOC patents': 570,
     'Active': '1994-2016',
     'Mainfieldsofinvention': 'Programming tools, DBX, Memory debuggers',
    },
    {'Inventor': 'David V. Horak',
     'NumberofPatentFamilies': 380,
     'Country': 'USA',
     'Number of INPADOC patents': 616,
     'Active': '1992-2016',
     'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'Kai-Li Jiang',
     'NumberofPatentFamilies': 379,
     'Country': 'China',
     'Number of INPADOC patents': 829,
     'Active': '2006-2016',

```

```

    'Mainfieldsofinvention': 'Carbon nanotubes and applications of
carbon nanotubes',
  },
  {'Inventor': 'Hans-Joachim Santel',
   'NumberofPatentFamilies': 377,
   'Country': 'Germany',
   'Number of INPADOC patents': 2623,
   'Active': '1986-2013',
   'Mainfieldsofinvention': 'Herbicides, Pesticides, Organic
chemistry',
  },
  {'Inventor': 'Xuemin (Sherman) Chen',
   'NumberofPatentFamilies': 377,
   'Country': 'USA',
   'Number of INPADOC patents': 1151,
   'Active': '1997-2017',
   'Mainfieldsofinvention': 'Computer networks, Integrated Circuits,
Signal Processing',
  },
  {'Inventor': 'George P. Liang',
   'NumberofPatentFamilies': 375,
   'Country': 'China',
   'Number of INPADOC patents': 508,
   'Active': '1983-2016',
   'Mainfieldsofinvention': 'Gas turbine cooling',
  },
  {'Inventor': 'Gisela Lorenz',
   'NumberofPatentFamilies': 374,
   'Country': 'Germany',
   'Number of INPADOC patents': 4155,
   'Active': '1990-2015',
   'Mainfieldsofinvention': 'Fungicides, Organic chemistry',
  },
  {'Inventor': 'Garry R. Jackson',
   'NumberofPatentFamilies': 367,
   'Country': 'Australia',
   'Number of INPADOC patents': 656,
   'Active': '2001-2016',
   'Mainfieldsofinvention': 'Printing, Mechanical',
  },
  {'Inventor': 'Paul W. Dent',
   'NumberofPatentFamilies': 362,
   'Country': 'USA',
   'Number of INPADOC patents': 2252,
   'Active': '1984-2015',
   'Mainfieldsofinvention': 'Wireless communications',
  },
  {'Inventor': 'George Westinghouse',
   'NumberofPatentFamilies': 361,
   'Country': 'USA',
   'Number of INPADOC patents': 'NA',
   'Active': '1846(b)-1914(d)',
   'Mainfieldsofinvention': 'Electric power, Electricity meter, Railway
air brake, Steam engines',
  },
  {'Inventor': 'Wael W. Diab',
   'NumberofPatentFamilies': 358,

```

```

        'Country': 'USA',
        'Number of INPADOC patents': 774,
        'Active': '2003-2016',
        'Mainfieldsofinvention': 'Computer networks',
    },
    {'Inventor': 'Devendra K. Sadana',
     'NumberofPatentFamilies': 356,
     'Country': 'India',
     'Number of INPADOC patents': 794,
     'Active': '1983-2016',
     'Mainfieldsofinvention': 'Solar cells, OLED, Integrated Circuits,
CMOS, DRAM, LEDs',
    },
    {'Inventor': 'Vincent J. Zimmer',
     'NumberofPatentFamilies': 354,
     'Country': 'USA',
     'Number of INPADOC patents': 972,
     'Active': '1999-2016',
     'Mainfieldsofinvention': 'Computer software and firmware',
    },
    {'Inventor': 'Robert R. Schmidt',
     'NumberofPatentFamilies': 350,
     'Country': 'Germany',
     'Number of INPADOC patents': 2467,
     'Active': '1971-2005',
     'Mainfieldsofinvention': 'Herbicides, Fungicides, Organic
chemistry',
    },
    {'Inventor': 'Norman M. Berry',
     'NumberofPatentFamilies': 347,
     'Country': 'Australia',
     'Number of INPADOC patents': 516,
     'Active': '2006-2016',
     'Mainfieldsofinvention': 'Printing, Mechanical',
    },
    {'Inventor': 'Chih-Chao Yang',
     'NumberofPatentFamilies': 345,
     'Country': 'USA',
     'Number of INPADOC patents': 690,
     'Active': '2003-2016',
     'Mainfieldsofinvention': 'Integrated Circuits',
    },
    {'Inventor': 'Gregory J. Boss',
     'NumberofPatentFamilies': 345,
     'Country': 'USA',
     'Number of INPADOC patents': 588,
     'Active': '2008-2016',
     'Mainfieldsofinvention': 'Various'
    },
    {'Inventor': 'Mark W. Kroll',
     'NumberofPatentFamilies': 343,
     'Country': 'USA',
     'Number of INPADOC patents': 460,
     'Active': '1987-2016',
     'Mainfieldsofinvention': 'Implantable medical devices',
    },
    {'Inventor': 'Brian M. O\'Connell',

```

```

        'NumberofPatentFamilies': 331,
        'Country': 'USA',
        'Number of INPADOC patents': 592,
        'Active': '2009-2016',
        'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'William Daniel Hillis',
     'NumberofPatentFamilies': 328,
     'Country': 'USA',
     'Number of INPADOC patents': 229,
     'Active': '1986-2016',
     'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'Brent A. Anderson',
     'NumberofPatentFamilies': 323,
     'Country': 'USA',
     'Number of INPADOC patents': 454,
     'Active': '2001-2016',
     'Mainfieldsofinvention': 'Semiconductor device fabrication,
Semiconductor memory, Semiconductor device',
    },
    {'Inventor': 'Jeffrey E. Stahmann',
     'NumberofPatentFamilies': 321,
     'Country': 'USA',
     'Number of INPADOC patents': 640,
     'Active': '1994-2016',
     'Mainfieldsofinvention': 'Medical devices',
    },
    {'Inventor': 'Carl J. Radens',
     'NumberofPatentFamilies': 317,
     'Country': 'USA',
     'Number of INPADOC patents': 636,
     'Active': '1994-2016',
     'Mainfieldsofinvention': 'Integrated Circuits, CMOS, DRAM,
Semiconductor device fabrication',
    },
    {'Inventor': 'Clifford A. Pickover',
     'NumberofPatentFamilies': 317,
     'Country': 'USA',
     'Number of INPADOC patents': 653,
     'Active': '1992-2016',
     'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'Liang Liu',
     'NumberofPatentFamilies': 310,
     'Country': 'China',
     'Number of INPADOC patents': 777,
     'Active': '2005-2016',
     'Mainfieldsofinvention': 'Carbon nanotubes and applications of
carbon nanotubes',
    },
    {'Inventor': 'Steven L. Teig',
     'NumberofPatentFamilies': 307,
     'Country': 'USA',
     'Number of INPADOC patents': 366,
     'Active': '1995-2016',
     'Mainfieldsofinvention': 'Integrated Circuits',
    },

```

```

    },
    {'Inventor': 'Victoria Smith',
     'NumberofPatentFamilies': 305,
     'Country': 'USA',
     'Number of INPADOC patents': 2040,
     'Active': '2006-2016',
     'Mainfieldsofinvention': 'Proteins, Antibodies',
    },
    {'Inventor': 'Robert G. LeTourneau',
     'NumberofPatentFamilies': 299,
     'Country': 'USA',
     'Number of INPADOC patents': 'NA',
     'Active': '1888(b)-1969(d)',
     'Mainfieldsofinvention': 'Earthworks (engineering), Heavy Equipment,
Industrial Machinery',
    },
    {'Inventor': 'William R. Tonti',
     'NumberofPatentFamilies': 291,
     'Country': 'USA',
     'Number of INPADOC patents': 441,
     'Active': '1994-2016',
     'Mainfieldsofinvention': 'Integrated Circuits, CMOS, DRAM,
Semiconductor device fabrication',
    },
    {'Inventor': 'Keith R. Walker',
     'NumberofPatentFamilies': 282,
     'Country': 'Saudi Arabia',
     'Number of INPADOC patents': 318,
     'Active': '2003-2016',
     'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'Victor S. Moore',
     'NumberofPatentFamilies': 280,
     'Country': 'USA',
     'Number of INPADOC patents': 428,
     'Active': '1982-2016',
     'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'Hanson S. Gifford III',
     'NumberofPatentFamilies': 276,
     'Country': 'USA',
     'Number of INPADOC patents': 795,
     'Active': '1987-2016',
     'Mainfieldsofinvention': 'Medical Devices',
    },
    {'Inventor': 'Daniel J. Winarski',
     'NumberofPatentFamilies': 275,
     'Country': 'USA',
     'Number of INPADOC patents': 506,
     'Active': '1982-2016',
     'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'Adam Heller',
     'NumberofPatentFamilies': 272,
     'Country': 'Romania',
     'Number of INPADOC patents': 711,
     'Active': '1968-2016',

```

```

    'Mainfieldsofinvention': 'Solar cells, Glucose meters, Lasers',
  },
  {'Inventor': 'Lisa Seacat DeLuca',
   'NumberofPatentFamilies': 271,
   'Country': 'USA',
   'Number of INPADOC patents': 385,
   'Active': '2009-2016',
   'Mainfieldsofinvention': 'Various',
  },
  {'Inventor': 'Brent Keeth',
   'NumberofPatentFamilies': 270,
   'Country': 'USA',
   'Number of INPADOC patents': 470,
   'Active': '1994-2016',
   'Mainfieldsofinvention': 'Integrated Circuits, CMOS, DRAM',
  },
  {'Inventor': 'Hartley Owen',
   'NumberofPatentFamilies': 267,
   'Country': 'USA',
   'Number of INPADOC patents': 751,
   'Active': '1976-2010',
   'Mainfieldsofinvention': 'Fluid catalytic cracking',
  },
  {'Inventor': 'Michael A. Rothman',
   'NumberofPatentFamilies': 256,
   'Country': 'USA',
   'Number of INPADOC patents': 687,
   'Active': '2001-2017',
   'Mainfieldsofinvention': 'Computer software and firmware',
  },
  {'Inventor': 'Yoshihiro Kikuchi',
   'NumberofPatentFamilies': 255,
   'Country': 'Japan',
   'Number of INPADOC patents': 1120,
   'Active': '1994-2015',
   'Mainfieldsofinvention': 'Video processing',
  },
  {'Inventor': 'Kulvir S. Bhogal',
   'NumberofPatentFamilies': 252,
   'Country': 'USA',
   'Number of INPADOC patents': 486,
   'Active': '2003-2016',
   'Mainfieldsofinvention': 'Various',
  },
  {'Inventor': 'Bengt Lindoff',
   'NumberofPatentFamilies': 248,
   'Country': 'Sweden',
   'Number of INPADOC patents': 1658,
   'Active': '2000-2017',
   'Mainfieldsofinvention': 'Wireless communications',
  },
  {'Inventor': 'Nobuyuki Taniguchi',
   'NumberofPatentFamilies': 245,
   'Country': 'Japan',
   'Number of INPADOC patents': 967,
   'Active': '1979-2015',
   'Mainfieldsofinvention': 'Cameras',
  },

```

```

    },
    {'Inventor': 'Dean L. Kamen',
     'NumberofPatentFamilies': 243,
     'Country': 'USA',
     'Number of INPADOC patents': 1186,
     'Active': '1979-2016',
     'Mainfieldsofinvention': 'Battery-powered electric vehicles, Medical
devices, Stirling engines, Water purification, Wheelchairs',
    },
    {'Inventor': 'Philip S. Yu',
     'NumberofPatentFamilies': 236,
     'Country': 'USA',
     'Number of INPADOC patents': 158,
     'Active': '1982-2016',
     'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'Rajiv V. Joshi',
     'NumberofPatentFamilies': 235,
     'Country': 'USA',
     'Number of INPADOC patents': 354,
     'Active': '1986-2016',
     'Mainfieldsofinvention': 'Electronics, analytics',
    },
    {'Inventor': 'Lawrence A. Clevenger',
     'NumberofPatentFamilies': 235,
     'Country': 'USA',
     'Number of INPADOC patents': 526,
     'Active': '1996-2017',
     'Mainfieldsofinvention': 'Semiconductor, Cognitive, Memory,
Security, Analytics',
    },
    {'Inventor': 'Johnny M. Shieh',
     'NumberofPatentFamilies': 231,
     'Country': 'USA',
     'Number of INPADOC patents': 444,
     'Active': '1996-2016',
     'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'Takeshi Chujoh',
     'NumberofPatentFamilies': 229,
     'Country': 'Japan',
     'Number of INPADOC patents': 1065,
     'Active': '1995-2016',
     'Mainfieldsofinvention': 'Video processing',
    },
    {'Inventor': 'Bran Ferren',
     'NumberofPatentFamilies': 225,
     'Country': 'USA',
     'Number of INPADOC patents': 589,
     'Active': '1986-2017',
     'Mainfieldsofinvention': 'Computers, Consumer Electronics, Optical
Systems, Medical, User Interfaces, Automotive',
    },
    {'Inventor': 'Paul Ian Mackey',
     'NumberofPatentFamilies': 220,
     'Country': 'Australia',
     'Number of INPADOC patents': 246,

```



```

        'Active': '2008-2016',
        'Mainfieldsofinvention': 'Printing, Mechanical',
    },
    {'Inventor': 'Louis Rosenberg',
     'NumberofPatentFamilies': 218,
     'Country': 'USA',
     'Number of INPADOC patents': 444,
     'Active': '1995-2016',
     'Mainfieldsofinvention': 'Augmented Reality, Virtual Reality, A.I.,
HCI',
    },
    {'Inventor': 'Thomas J. Kennedy III',
     'NumberofPatentFamilies': 218,
     'Country': 'USA',
     'Number of INPADOC patents': 513,
     'Active': '1992-2016',
     'Mainfieldsofinvention': 'Sporting Goods, Wind Turbines',
    },
    {'Inventor': 'Gerald F. McBrearty',
     'NumberofPatentFamilies': 213,
     'Country': 'USA',
     'Number of INPADOC patents': 387,
     'Active': '1997-2016',
     'Mainfieldsofinvention': 'Various',
    },
    {'Inventor': 'Esmael H. Dinan',
     'NumberofPatentFamilies': 208,
     'Country': 'USA',
     'Number of INPADOC patents': 344,
     'Active': '2000-2017',
     'Mainfieldsofinvention': 'Communication Networks',
    },
    {'Inventor': 'Imad Libbus',
     'NumberofPatentFamilies': 207,
     'Country': 'USA',
     'Number of INPADOC patents': 472,
     'Active': '2007-2017',
     'Mainfieldsofinvention': 'Medical devices',
    },
    {'Inventor': 'Hiroshi (You) Yoshioka',
     'NumberofPatentFamilies': 205,
     'Country': 'Japan',
     'Number of INPADOC patents': 181,
     'Active': '1997-2015',
     'Mainfieldsofinvention': 'Cameras',
    },
    {'Inventor': 'Patrick B. Usoro',
     'NumberofPatentFamilies': 205,
     'Country': 'USA',
     'Number of INPADOC patents': 343,
     'Active': '1999-2016',
     'Mainfieldsofinvention': 'Transmissions, Hybrid Powertrains, Vehicle
Thermal Management',
    },
    {'Inventor': 'Gregory McAvoy',
     'NumberofPatentFamilies': 205,
     'Country': 'Australia',

```

```

        'Number of INPADOC patents': 433,
        'Active': '2003-2014',
        'Mainfieldsofinvention': 'Printing, MEMS',
    },
    {'Inventor': 'Sebastian T Ventrone',
     'NumberofPatentFamilies': 204,
     'Country': 'USA',
     'Number of INPADOC patents': 283,
     'Active': '1989-2017',
     'Mainfieldsofinvention': 'Semiconductor, Logic, Architecture',
    },
    {'Inventor': 'Dorin Comaniciu',
     'NumberofPatentFamilies': 200,
     'Country': 'USA',
     'Number of INPADOC patents': 452,
     'Active': '2003-2017',
     'Mainfieldsofinvention': 'Machine Intelligence, Medical Imaging,
Image-Guided Surgery, Computer Vision',
    }
];
export interface ColumnSpanDataType {EmployeeID}: number;
    EmployeeName: string;
    '9:00': string;
    '9:30': string;
    '10:00': string;
    '10:30': string;
    '11:00': string;
    '11:30': string;
    '12:00': string;
    '12:30': string;
    '1:00': string;
    '1:30': string;
    '2:00': string;
    '2:30': string;
    '3:00': string;
    '3:30': string;
    '4:00': string;
    '4:30': string;
    '5:00': string;
}
export let columnSpanData: ColumnSpanDataType[] = [
    {EmployeeID}: 10001,
    EmployeeName: 'Davolio',
    '9:00': 'Analysis Tasks',
    '9:30': 'Analysis Tasks',
    '10:00': 'Team Meeting',
    '10:30': 'Testing',
    '11:00': 'Development',
    '11:30': 'Development',
    '12:00': 'Development',
    '12:30': 'Support',
    '1:00': 'Lunch Break',
    '1:30': 'Lunch Break',
    '2:00': 'Lunch Break',
    '2:30': 'Testing',
    '3:00': 'Testing',
    '3:30': 'Development',

```

```

        '4:00': 'Conference',
        '4:30': 'Team Meeting',
        '5:00': 'Team Meeting'
    },
    {EmployeeID: 10002,
      EmployeeName: 'Buchanan',
      '9:00': 'Task Assign',
      '9:30': 'Support',
      '10:00': 'Support',
      '10:30': 'Support',
      '11:00': 'Testing',
      '11:30': 'Testing',
      '12:00': 'Testing',
      '12:30': 'Testing',
      '1:00': 'Lunch Break',
      '1:30': 'Lunch Break',
      '2:00': 'Lunch Break',
      '2:30': 'Development',
      '3:00': 'Development',
      '3:30': 'Check Mail',
      '4:00': 'Check Mail',
      '4:30': 'Team Meeting',
      '5:00': 'Team Meeting'
    },
    {EmployeeID: 10003,
      EmployeeName: 'Fuller',
      '9:00': 'Check Mail',
      '9:30': 'Check Mail',
      '10:00': 'Check Mail',
      '10:30': 'Analysis Tasks',
      '11:00': 'Analysis Tasks',
      '11:30': 'Support',
      '12:00': 'Support',
      '12:30': 'Support',
      '1:00': 'Lunch Break',
      '1:30': 'Lunch Break',
      '2:00': 'Lunch Break',
      '2:30': 'Development',
      '3:00': 'Development',
      '3:30': 'Team Meeting',
      '4:00': 'Team Meeting',
      '4:30': 'Development',
      '5:00': 'Development'
    },
    {EmployeeID: 10004,
      EmployeeName: 'Leverling',
      '9:00': 'Testing',
      '9:30': 'Check Mail',
      '10:00': 'Check Mail',
      '10:30': 'Support',
      '11:00': 'Testing',
      '11:30': 'Testing',
      '12:00': 'Testing',
      '12:30': 'Testing',
      '1:00': 'Lunch Break',
      '1:30': 'Lunch Break',
      '2:00': 'Lunch Break',

```

```

        '2:30': 'Development',
        '3:00': 'Development',
        '3:30': 'Check Mail',
        '4:00': 'Conference',
        '4:30': 'Conference',
        '5:00': 'Team Meeting'
    },
    {EmployeeID}: 10005,
    EmployeeName: 'Peacock',
    '9:00': 'Task Assign',
    '9:30': 'Task Assign',
    '10:00': 'Task Assign',
    '10:30': 'Task Assign',
    '11:00': 'Check Mail',
    '11:30': 'Support',
    '12:00': 'Support',
    '12:30': 'Support',
    '1:00': 'Lunch Break',
    '1:30': 'Lunch Break',
    '2:00': 'Lunch Break',
    '2:30': 'Development',
    '3:00': 'Development',
    '3:30': 'Team Meeting',
    '4:00': 'Team Meeting',
    '4:30': 'Testing',
    '5:00': 'Testing'
},
    {EmployeeID}: 10006,
    EmployeeName: 'Janet',
    '9:00': 'Testing',
    '9:30': 'Testing',
    '10:00': 'Support',
    '10:30': 'Support',
    '11:00': 'Support',
    '11:30': 'Team Meeting',
    '12:00': 'Team Meeting',
    '12:30': 'Team Meeting',
    '1:00': 'Lunch Break',
    '1:30': 'Lunch Break',
    '2:00': 'Lunch Break',
    '2:30': 'Development',
    '3:00': 'Development',
    '3:30': 'Team Meeting',
    '4:00': 'Team Meeting',
    '4:30': 'Development',
    '5:00': 'Development'
},
    {EmployeeID}: 10007,
    EmployeeName: 'Suyama',
    '9:00': 'Analysis Tasks',
    '9:30': 'Analysis Tasks',
    '10:00': 'Testing',
    '10:30': 'Development',
    '11:00': 'Development',
    '11:30': 'Testing',
    '12:00': 'Testing',
    '12:30': 'Testing',

```

```

        '1:00': 'Lunch Break',
        '1:30': 'Lunch Break',
        '2:00': 'Lunch Break',
        '2:30': 'Support',
        '3:00': 'Build',
        '3:30': 'Build',
        '4:00': 'Check Mail',
        '4:30': 'Check Mail',
        '5:00': 'Check Mail'
    },
    {EmployeeID}: 10008,
        EmployeeName: 'Robert',
        '9:00': 'Task Assign',
        '9:30': 'Task Assign',
        '10:00': 'Task Assign',
        '10:30': 'Development',
        '11:00': 'Development',
        '11:30': 'Development',
        '12:00': 'Testing',
        '12:30': 'Support',
        '1:00': 'Lunch Break',
        '1:30': 'Lunch Break',
        '2:00': 'Lunch Break',
        '2:30': 'Check Mail',
        '3:00': 'Check Mail',
        '3:30': 'Check Mail',
        '4:00': 'Team Meeting',
        '4:30': 'Team Meeting',
        '5:00': 'Build'
    },
    {EmployeeID}: 10009,
        EmployeeName: 'Andrew',
        '9:00': 'Check Mail',
        '9:30': 'Team Meeting',
        '10:00': 'Team Meeting',
        '10:30': 'Support',
        '11:00': 'Testing',
        '11:30': 'Development',
        '12:00': 'Development',
        '12:30': 'Development',
        '1:00': 'Lunch Break',
        '1:30': 'Lunch Break',
        '2:00': 'Lunch Break',
        '2:30': 'Check Mail',
        '3:00': 'Check Mail',
        '3:30': 'Check Mail',
        '4:00': 'Team Meeting',
        '4:30': 'Development',
        '5:00': 'Development'
    },
    {EmployeeID}: 10010,
        EmployeeName: 'Michael',
        '9:00': 'Task Assign',
        '9:30': 'Task Assign',
        '10:00': 'Task Assign',
        '10:30': 'Analysis Tasks',
        '11:00': 'Analysis Tasks',

```

```

    '11:30': 'Development',
    '12:00': 'Development',
    '12:30': 'Development',
    '1:00': 'Lunch Break',
    '1:30': 'Lunch Break',
    '2:00': 'Lunch Break',
    '2:30': 'Testing',
    '3:00': 'Testing',
    '3:30': 'Testing',
    '4:00': 'Build',
    '4:30': 'Build',
    '5:00': 'Build'
  }
];</>;

```

DATASOURCE.TS

```

export let productData: Object[] = [
  {
    'ProductID': 1,
    'ProductName': 'Chai',
    'SupplierID': 1,
    'QuantityPerUnit': '10 boxes x 20 bags',
    'UnitPrice': 18.00,
    'UnitsInStock': 39,
    'Discontinued': true
  },
  {
    'ProductID': 2,
    'ProductName': 'Chang',
    'SupplierID': 1,
    'QuantityPerUnit': '24 - 12 oz bottles',
    'UnitPrice': 19.00,
    'UnitsInStock': 17,
    'Discontinued': true
  },
  {
    'ProductID': 3,
    'ProductName': 'Aniseed Syrup',
    'SupplierID': 1,
    'QuantityPerUnit': '12 - 550 ml bottles',
    'UnitPrice': 10.00,
    'UnitsInStock': 13,
    'Discontinued': true
  },
  {
    'ProductID': 4,
    'ProductName': 'Chef Anton\'s Cajun Seasoning',
    'SupplierID': 2,
    'QuantityPerUnit': '48 - 6 oz jars',
    'UnitPrice': 22.00,
    'UnitsInStock': 53,
    'Discontinued': true
  },
  {
    'ProductID': 5,

```

```

    'ProductName': 'Chef Anton\'s Gumbo Mix',
    'SupplierID': 2,
    'QuantityPerUnit': '36 boxes',
    'UnitPrice': 21.35,
    'UnitsInStock': 0,
    'Discontinued': true
  },
  {
    'ProductID': 6,
    'ProductName': 'Grandma\'s Boysenberry Spread',
    'SupplierID': 3,
    'QuantityPerUnit': '12 - 8 oz jars',
    'UnitPrice': 25.00,
    'UnitsInStock': 120,
    'Discontinued': false
  },
  {
    'ProductID': 7,
    'ProductName': 'Uncle Bob\'s Organic Dried Pears',
    'SupplierID': 3,
    'QuantityPerUnit': '12 - 1 lb pkgs.',
    'UnitPrice': 30.00,
    'UnitsInStock': 15,
    'Discontinued': false
  },
  {
    'ProductID': 8,
    'ProductName': 'Northwoods Cranberry Sauce',
    'SupplierID': 3,
    'QuantityPerUnit': '12 - 12 oz jars',
    'UnitPrice': 40.00,
    'UnitsInStock': 6,
    'Discontinued': false
  },
  {
    'ProductID': 9,
    'ProductName': 'Mishi Kobe Niku',
    'SupplierID': 4,
    'QuantityPerUnit': '18 - 500 g pkgs.',
    'UnitPrice': 97.00,
    'UnitsInStock': 29,
    'Discontinued': true
  },
  {
    'ProductID': 10,
    'ProductName': 'Ikura',
    'SupplierID': 4,
    'QuantityPerUnit': '12 - 200 ml jars',
    'UnitPrice': 31.00,
    'UnitsInStock': 31,
    'Discontinued': false
  },
  {
    'ProductID': 11,
    'ProductName': 'Queso Cabrales',
    'SupplierID': 5,
    'QuantityPerUnit': '1 kg pkg.',

```

```
'UnitPrice': 21.00,  
'UnitsInStock': 22,  
'Discontinued': false  
},  
{  
  'ProductID': 12,  
  'ProductName': 'Queso Manchego La Pastora',  
  'SupplierID': 5,  
  'QuantityPerUnit': '10 - 500 g pkgs.',  
  'UnitPrice': 38.00,  
  'UnitsInStock': 86,  
  'Discontinued': false  
},  
{  
  'ProductID': 13,  
  'ProductName': 'Konbu',  
  'SupplierID': 6,  
  'QuantityPerUnit': '2 kg box',  
  'UnitPrice': 6.00,  
  'UnitsInStock': 24,  
  'Discontinued': true  
},  
{  
  'ProductID': 14,  
  'ProductName': 'Tofu',  
  'SupplierID': 6,  
  'QuantityPerUnit': '40 - 100 g pkgs.',  
  'UnitPrice': 23.25,  
  'UnitsInStock': 35,  
  'Discontinued': true  
},  
{  
  'ProductID': 15,  
  'ProductName': 'Genen Shouyu',  
  'SupplierID': 6,  
  'QuantityPerUnit': '24 - 250 ml bottles',  
  'UnitPrice': 15.50,  
  'UnitsInStock': 39,  
  'Discontinued': true  
},  
{  
  'ProductID': 16,  
  'ProductName': 'Pavlova',  
  'SupplierID': 7,  
  'QuantityPerUnit': '32 - 500 g boxes',  
  'UnitPrice': 17.45,  
  'UnitsInStock': 29,  
  'Discontinued': true  
},  
{  
  'ProductID': 17,  
  'ProductName': 'Alice Mutton',  
  'SupplierID': 7,  
  'QuantityPerUnit': '20 - 1 kg tins',  
  'UnitPrice': 39.00,  
  'UnitsInStock': 0,  
  'Discontinued': true
```



```

    },
    {
        'ProductID': 18,
        'ProductName': 'Carnarvon Tigers',
        'SupplierID': 7,
        'QuantityPerUnit': '16 kg pkg.',
        'UnitPrice': 62.50,
        'UnitsInStock': 42,
        'Discontinued': false
    },
    {
        'ProductID': 19,
        'ProductName': 'Teatime Chocolate Biscuits',
        'SupplierID': 8,
        'QuantityPerUnit': '10 boxes x 12 pieces',
        'UnitPrice': 9.20,
        'UnitsInStock': 25,
        'Discontinued': false
    },
    {
        'ProductID': 20,
        'ProductName': 'Sir Rodney\'s Marmalade',
        'SupplierID': 8,
        'QuantityPerUnit': '30 gift boxes',
        'UnitPrice': 81.00,
        'UnitsInStock': 40,
        'Discontinued': false
    },
    {
        'ProductID': 21,
        'ProductName': 'Sir Rodney\'s Scones',
        'SupplierID': 8,
        'QuantityPerUnit': '24 pkgs. x 4 pieces',
        'UnitPrice': 10.00,
        'UnitsInStock': 3,
        'Discontinued': false
    },
    {
        'ProductID': 22,
        'ProductName': 'Gustaf\'s Knäckebröd',
        'SupplierID': 9,
        'QuantityPerUnit': '24 - 500 g pkgs.',
        'UnitPrice': 21.00,
        'UnitsInStock': 104,
        'Discontinued': false
    },
    {
        'ProductID': 23,
        'ProductName': 'Tunnbröd',
        'SupplierID': 9,
        'QuantityPerUnit': '12 - 250 g pkgs.',
        'UnitPrice': 9.00,
        'UnitsInStock': 61,
        'Discontinued': false
    },
    {
        'ProductID': 24,

```

```

    'ProductName': 'Guaraná Fantástica',
    'SupplierID': 10,
    'QuantityPerUnit': '12 - 355 ml cans',
    'UnitPrice': 4.50,
    'UnitsInStock': 20,
    'Discontinued': true
  },
  {
    'ProductID': 25,
    'ProductName': 'NuNuCa Nuß-Nougat-Creme',
    'SupplierID': 11,
    'QuantityPerUnit': '20 - 450 g glasses',
    'UnitPrice': 14.00,
    'UnitsInStock': 76,
    'Discontinued': false
  },
  {
    'ProductID': 26,
    'ProductName': 'Gumbär Gummibärchen',
    'SupplierID': 11,
    'QuantityPerUnit': '100 - 250 g bags',
    'UnitPrice': 31.23,
    'UnitsInStock': 15,
    'Discontinued': true
  },
  {
    'ProductID': 27,
    'ProductName': 'Schoggi Schokolade',
    'SupplierID': 11,
    'QuantityPerUnit': '100 - 100 g pieces',
    'UnitPrice': 43.90,
    'UnitsInStock': 49,
    'Discontinued': true
  },
  {
    'ProductID': 28,
    'ProductName': 'Rössle Sauerkraut',
    'SupplierID': 12,
    'QuantityPerUnit': '25 - 825 g cans',
    'UnitPrice': 45.60,
    'UnitsInStock': 26,
    'Discontinued': true
  },
  {
    'ProductID': 29,
    'ProductName': 'Thüringer Rostbratwurst',
    'SupplierID': 12,
    'QuantityPerUnit': '50 bags x 30 sausgs.',
    'UnitPrice': 123.79,
    'UnitsInStock': 0,
    'Discontinued': true
  },
  {
    'ProductID': 30,
    'ProductName': 'Nord-Ost Matjeshering',
    'SupplierID': 13,
    'QuantityPerUnit': '10 - 200 g glasses',

```

```
'UnitPrice': 25.89,  
'UnitsInStock': 10,  
'Discontinued': true  
},  
{  
  'ProductID': 31,  
  'ProductName': 'Gorgonzola Telino',  
  'SupplierID': 14,  
  'QuantityPerUnit': '12 - 100 g pkgs',  
  'UnitPrice': 12.50,  
  'UnitsInStock': 0,  
  'Discontinued': true  
},  
{  
  'ProductID': 32,  
  'ProductName': 'Mascarpone Fabioli',  
  'SupplierID': 14,  
  'QuantityPerUnit': '24 - 200 g pkgs.',  
  'UnitPrice': 32.00,  
  'UnitsInStock': 9,  
  'Discontinued': false  
},  
{  
  'ProductID': 33,  
  'ProductName': 'Geitost',  
  'SupplierID': 15,  
  'QuantityPerUnit': '500 g',  
  'UnitPrice': 2.50,  
  'UnitsInStock': 112,  
  'Discontinued': false  
},  
{  
  'ProductID': 34,  
  'ProductName': 'Sasquatch Ale',  
  'SupplierID': 16,  
  'QuantityPerUnit': '24 - 12 oz bottles',  
  'UnitPrice': 14.00,  
  'UnitsInStock': 111,  
  'Discontinued': false  
},  
{  
  'ProductID': 35,  
  'ProductName': 'Steeleye Stout',  
  'SupplierID': 16,  
  'QuantityPerUnit': '24 - 12 oz bottles',  
  'UnitPrice': 18.00,  
  'UnitsInStock': 20,  
  'Discontinued': false  
},  
{  
  'ProductID': 36,  
  'ProductName': 'Inlagd Sill',  
  'SupplierID': 17,  
  'QuantityPerUnit': '24 - 250 g jars',  
  'UnitPrice': 19.00,  
  'UnitsInStock': 112,  
  'Discontinued': false
```

```

    },
    {
        'ProductID': 37,
        'ProductName': 'Gravad lax',
        'SupplierID': 17,
        'QuantityPerUnit': '12 - 500 g pkgs.',
        'UnitPrice': 26.00,
        'UnitsInStock': 11,
        'Discontinued': false
    },
    {
        'ProductID': 38,
        'ProductName': 'Côte de Blaye',
        'SupplierID': 18,
        'QuantityPerUnit': '12 - 75 cl bottles',
        'UnitPrice': 263.50,
        'UnitsInStock': 17,
        'Discontinued': false
    },
    {
        'ProductID': 39,
        'ProductName': 'Chartreuse verte',
        'SupplierID': 18,
        'QuantityPerUnit': '750 cc per bottle',
        'UnitPrice': 18.00,
        'UnitsInStock': 69,
        'Discontinued': true
    },
    {
        'ProductID': 40,
        'ProductName': 'Boston Crab Meat',
        'SupplierID': 19,
        'QuantityPerUnit': '24 - 4 oz tins',
        'UnitPrice': 18.40,
        'UnitsInStock': 123,
        'Discontinued': true
    },
    {
        'ProductID': 41,
        'ProductName': 'Jack\'s New England Clam Chowder',
        'SupplierID': 19,
        'QuantityPerUnit': '12 - 12 oz cans',
        'UnitPrice': 9.65,
        'UnitsInStock': 85,
        'Discontinued': false
    },
    {
        'ProductID': 42,
        'ProductName': 'Singaporean Hokkien Fried Mee',
        'SupplierID': 20,
        'QuantityPerUnit': '32 - 1 kg pkgs.',
        'UnitPrice': 14.00,
        'UnitsInStock': 26,
        'Discontinued': true
    },
    {
        'ProductID': 43,

```

```

        'ProductName': 'Ipoh Coffee',
        'SupplierID': 20,
        'QuantityPerUnit': '16 - 500 g tins',
        'UnitPrice': 46.00,
        'UnitsInStock': 17,
        'Discontinued': false
    },
    {
        'ProductID': 44,
        'ProductName': 'Gula Malacca',
        'SupplierID': 20,
        'QuantityPerUnit': '20 - 2 kg bags',
        'UnitPrice': 19.45,
        'UnitsInStock': 27,
        'Discontinued': false
    },
    {
        'ProductID': 45,
        'ProductName': 'Rogede sild',
        'SupplierID': 21,
        'QuantityPerUnit': '1k pkg.',
        'UnitPrice': 9.50,
        'UnitsInStock': 5,
        'Discontinued': true
    },
    {
        'ProductID': 46,
        'ProductName': 'Spegesild',
        'SupplierID': 21,
        'QuantityPerUnit': '4 - 450 g glasses',
        'UnitPrice': 12.00,
        'UnitsInStock': 95,
        'Discontinued': true
    },
    {
        'ProductID': 47,
        'ProductName': 'Zaanse koeken',
        'SupplierID': 22,
        'QuantityPerUnit': '10 - 4 oz boxes',
        'UnitPrice': 9.50,
        'UnitsInStock': 36,
        'Discontinued': true
    },
    {
        'ProductID': 48,
        'ProductName': 'Chocolade',
        'SupplierID': 22,
        'QuantityPerUnit': '10 pkgs.',
        'UnitPrice': 12.75,
        'UnitsInStock': 15,
        'Discontinued': true
    },
    {
        'ProductID': 49,
        'ProductName': 'Maxilaku',
        'SupplierID': 23,
        'QuantityPerUnit': '24 - 50 g pkgs.',

```

```

      'UnitPrice': 20.00,
      'UnitsInStock': 10,
      'Discontinued': false
    },
    {
      'ProductID': 50,
      'ProductName': 'Valkoinen suklaa',
      'SupplierID': 23,
      'QuantityPerUnit': '12 - 100 g bars',
      'UnitPrice': 16.25,
      'UnitsInStock': 65,
      'Discontinued': false
    },
    {
      'ProductID': 51,
      'ProductName': 'Manjimup Dried Apples',
      'SupplierID': 24,
      'QuantityPerUnit': '50 - 300 g pkgs.',
      'UnitPrice': 53.00,
      'UnitsInStock': 20,
      'Discontinued': false
    },
    {
      'ProductID': 52,
      'ProductName': 'Filo Mix',
      'SupplierID': 24,
      'QuantityPerUnit': '16 - 2 kg boxes',
      'UnitPrice': 7.00,
      'UnitsInStock': 38,
      'Discontinued': true
    },
    {
      'ProductID': 53,
      'ProductName': 'Perth Pasties',
      'SupplierID': 24,
      'QuantityPerUnit': '48 pieces',
      'UnitPrice': 32.80,
      'UnitsInStock': 0,
      'Discontinued': true
    },
    {
      'ProductID': 54,
      'ProductName': 'Tourtière',
      'SupplierID': 25,
      'QuantityPerUnit': '16 pies',
      'UnitPrice': 7.45,
      'UnitsInStock': 21,
      'Discontinued': true
    },
    {
      'ProductID': 55,
      'ProductName': 'Pâté chinois',
      'SupplierID': 25,
      'QuantityPerUnit': '24 boxes x 2 pies',
      'UnitPrice': 24.00,
      'UnitsInStock': 115,
      'Discontinued': true
    }
  ]

```

```

    },
    {
        'ProductID': 56,
        'ProductName': 'Gnocchi di nonna Alice',
        'SupplierID': 26,
        'QuantityPerUnit': '24 - 250 g pkgs.',
        'UnitPrice': 38.00,
        'UnitsInStock': 21,
        'Discontinued': false
    },
    {
        'ProductID': 57,
        'ProductName': 'Ravioli Angelo',
        'SupplierID': 26,
        'QuantityPerUnit': '24 - 250 g pkgs.',
        'UnitPrice': 19.50,
        'UnitsInStock': 36,
        'Discontinued': false
    },
    {
        'ProductID': 58,
        'ProductName': 'Escargots de Bourgogne',
        'SupplierID': 27,
        'QuantityPerUnit': '24 pieces',
        'UnitPrice': 13.25,
        'UnitsInStock': 62,
        'Discontinued': false
    },
    {
        'ProductID': 59,
        'ProductName': 'Raclette Courdavault',
        'SupplierID': 28,
        'QuantityPerUnit': '5 kg pkg.',
        'UnitPrice': 55.00,
        'UnitsInStock': 79,
        'Discontinued': false
    },
    {
        'ProductID': 60,
        'ProductName': 'Camembert Pierrot',
        'SupplierID': 28,
        'QuantityPerUnit': '15 - 300 g rounds',
        'UnitPrice': 34.00,
        'UnitsInStock': 19,
        'Discontinued': false
    },
    {
        'ProductID': 61,
        'ProductName': 'Sirop d\'érable',
        'SupplierID': 29,
        'QuantityPerUnit': '24 - 500 ml bottles',
        'UnitPrice': 28.50,
        'UnitsInStock': 113,
        'Discontinued': false
    },
    {
        'ProductID': 62,

```

```

    'ProductName': 'Tarte au sucre',
    'SupplierID': 29,
    'QuantityPerUnit': '48 pies',
    'UnitPrice': 49.30,
    'UnitsInStock': 17,
    'Discontinued': false
  },
  {
    'ProductID': 63,
    'ProductName': 'Vegie-spread',
    'SupplierID': 7,
    'QuantityPerUnit': '15 - 625 g jars',
    'UnitPrice': 43.90,
    'UnitsInStock': 24,
    'Discontinued': true
  },
  {
    'ProductID': 64,
    'ProductName': 'Wimmers gute Semmelknödel',
    'SupplierID': 12,
    'QuantityPerUnit': '20 bags x 4 pieces',
    'UnitPrice': 33.25,
    'UnitsInStock': 22,
    'Discontinued': true
  },
  {
    'ProductID': 65,
    'ProductName': 'Louisiana Fiery Hot Pepper Sauce',
    'SupplierID': 2,
    'QuantityPerUnit': '32 - 8 oz bottles',
    'UnitPrice': 21.05,
    'UnitsInStock': 76,
    'Discontinued': true
  },
  {
    'ProductID': 66,
    'ProductName': 'Louisiana Hot Spiced Okra',
    'SupplierID': 2,
    'QuantityPerUnit': '24 - 8 oz jars',
    'UnitPrice': 17.00,
    'UnitsInStock': 4,
    'Discontinued': false
  },
  {
    'ProductID': 67,
    'ProductName': 'Laughing Lumberjack Lager',
    'SupplierID': 16,
    'QuantityPerUnit': '24 - 12 oz bottles',
    'UnitPrice': 14.00,
    'UnitsInStock': 52,
    'Discontinued': false
  },
  {
    'ProductID': 68,
    'ProductName': 'Scottish Longbreads',
    'SupplierID': 8,
    'QuantityPerUnit': '10 boxes x 8 pieces',

```



```
'UnitPrice': 12.50,  
'UnitsInStock': 6,  
'Discontinued': false  
},  
{  
  'ProductID': 69,  
  'ProductName': 'Gudbrandsdalsost',  
  'SupplierID': 15,  
  'QuantityPerUnit': '10 kg pkg.',  
  'UnitPrice': 36.00,  
  'UnitsInStock': 26,  
  'Discontinued': false  
},  
{  
  'ProductID': 70,  
  'ProductName': 'Outback Lager',  
  'SupplierID': 7,  
  'QuantityPerUnit': '24 - 355 ml bottles',  
  'UnitPrice': 15.00,  
  'UnitsInStock': 15,  
  'Discontinued': false  
},  
{  
  'ProductID': 71,  
  'ProductName': 'Flotemysost',  
  'SupplierID': 15,  
  'QuantityPerUnit': '10 - 500 g pkgs.',  
  'UnitPrice': 21.50,  
  'UnitsInStock': 26,  
  'Discontinued': true  
},  
{  
  'ProductID': 72,  
  'ProductName': 'Mozzarella di Giovanni',  
  'SupplierID': 14,  
  'QuantityPerUnit': '24 - 200 g pkgs.',  
  'UnitPrice': 34.80,  
  'UnitsInStock': 14,  
  'Discontinued': true  
},  
{  
  'ProductID': 73,  
  'ProductName': 'Röd Kaviar',  
  'SupplierID': 17,  
  'QuantityPerUnit': '24 - 150 g jars',  
  'UnitPrice': 15.00,  
  'UnitsInStock': 101,  
  'Discontinued': true  
},  
{  
  'ProductID': 74,  
  'ProductName': 'Longlife Tofu',  
  'SupplierID': 4,  
  'QuantityPerUnit': '5 kg pkg.',  
  'UnitPrice': 10.00,  
  'UnitsInStock': 4,  
  'Discontinued': true
```

```

    },
    {
        'ProductID': 75,
        'ProductName': 'Rhönbräu Klosterbier',
        'SupplierID': 12,
        'QuantityPerUnit': '24 - 0.5 l bottles',
        'UnitPrice': 7.75,
        'UnitsInStock': 125,
        'Discontinued': true
    },
    {
        'ProductID': 76,
        'ProductName': 'Lakkalikööri',
        'SupplierID': 23,
        'QuantityPerUnit': '500 ml',
        'UnitPrice': 18.00,
        'UnitsInStock': 57,
        'Discontinued': true
    },
    {
        'ProductID': 77,
        'ProductName': 'Original Frankfurter grüne Soße',
        'SupplierID': 12,
        'QuantityPerUnit': '12 boxes',
        'UnitPrice': 13.00,
        'UnitsInStock': 32,
        'Discontinued': false
    }
    ]];
export let employeeData: Object[] = [{
    'EmployeeID': 1,
    'LastName': 'Davolio',
    'FirstName': 'Nancy',
    'Title': 'Sales Representative',
    'TitleOfCourtesy': 'Ms.',
    'BirthDate': new Date(-664743600000),
    'HireDate': new Date(704692800000),
    'Address': '507 - 20th Ave. E.\r\nApt. 2A',
    'City': 'Seattle',
    'Region': 'WA',
    'PostalCode': '98122',
    'Country': 'USA',
    'HomePhone': '(206) 555-9857',
    'Extension': '5467',
    'Photo': { 'Length': 21626 },
    'Notes': 'Education includes a BA in psychology from Colorado State University in 1970. She also completed\'
    \'The Art of the Cold Call.\' Nancy is a member of Toastmasters International.',
    'ReportsTo': 2,
    'PhotoPath': 'http://accweb/emmployees/davolio.bmp'
},
{
    'EmployeeID': 2,
    'LastName': 'Fuller',
    'FirstName': 'Andrew',
    'Title': 'Vice President, Sales',
    'TitleOfCourtesy': 'Dr.',

```

```

    'BirthDate': new Date(-563828400000),
    'HireDate': new Date(713764800000),
    'Address': '908 W. Capital Way',
    'City': 'Tacoma',
    'Region': 'WA',
    'PostalCode': '98401',
    'Country': 'USA',
    'HomePhone': '(206) 555-9482',
    'Extension': '3457',
    'Photo': { 'Length': 21626 },
    'Notes': 'Andrew received his BTS commercial in 1974 and a Ph.D. in
international marketing from the University of \
    Dallas in 1981. He is fluent in French and Italian and reads German.
He joined the company as a sales representative, \
    was promoted to sales manager in January 1992 and to vice president of
sales in March 1993. Andrew is a member of the \
    Sales Management Roundtable, the Seattle Chamber of Commerce, and the
Pacific Rim Importers Association.',
    'ReportsTo': 0,
    'PhotoPath': 'http://accweb/emmployees/fuller.bmp'
},
{
    'EmployeeID': 3,
    'LastName': 'Leverling',
    'FirstName': 'Janet',
    'Title': 'Sales Representative',
    'TitleOfCourtesy': 'Ms.',
    'BirthDate': new Date(-200088000000),
    'HireDate': new Date(702104400000),
    'Address': '722 Moss Bay Blvd.',
    'City': 'Kirkland',
    'Region': 'WA',
    'PostalCode': '98033',
    'Country': 'USA',
    'HomePhone': '(206) 555-3412',
    'Extension': '3355',
    'Photo': { 'Length': 21722 },
    'Notes': 'Janet has a BS degree in chemistry from Boston College (1984).
\
    She has also completed a certificate program in food retailing
management.\
    Janet was hired as a sales associate in 1991 and promoted to sales
representative in February 1992.',
    'ReportsTo': 2,
    'PhotoPath': 'http://accweb/emmployees/leverling.bmp'
},
{
    'EmployeeID': 4,
    'LastName': 'Peacock',
    'FirstName': 'Margaret',
    'Title': 'Sales Representative',
    'TitleOfCourtesy': 'Mrs.',
    'BirthDate': new Date(-1018814400000),
    'HireDate': new Date(736401600000),
    'Address': '4110 Old Redmond Rd.',
    'City': 'Redmond',
    'Region': 'WA',

```

```

    'PostalCode': '98052',
    'Country': 'USA',
    'HomePhone': '(206) 555-8122',
    'Extension': '5176',
    'Photo': { 'Length': 21626 },
    'Notes': 'Margaret holds a BA in English literature from Concordia
College (1958) and an MA from the American \
Institute of Culinary Arts (1966). She was assigned to the London
office temporarily from July through November 1992.',
    'ReportsTo': 2,
    'PhotoPath': 'http://accweb/emmployees/peacock.bmp'
},
{
    'EmployeeID': 5,
    'LastName': 'Buchanan',
    'FirstName': 'Steven',
    'Title': 'Sales Manager',
    'TitleOfCourtesy': 'Mr.',
    'BirthDate': new Date(-468010800000),
    'HireDate': new Date(750830400000),
    'Address': '14 Garrett Hill',
    'City': 'London',
    'Region': null,
    'PostalCode':
    'SW1 8JR',
    'Country': 'UK',
    'HomePhone': '(71) 555-4848',
    'Extension': '3453',
    'Photo': { 'Length': 21626 },
    'Notes': 'Steven Buchanan graduated from St. Andrews University,
Scotland, with a BSC degree in 1976. Upon joining the company as \
a sales representative in 1992, he spent 6 months in an orientation
program at the Seattle office and then returned to his permanent \
post in London. He was promoted to sales manager in March 1993. Mr.
Buchanan has completed the courses \'Successful \
Telemarketing\' and \'International Sales Management.\' He is fluent in
French.',
    'ReportsTo': 2,
    'PhotoPath': 'http://accweb/emmployees/buchanan.bmp'
},
{
    'EmployeeID': 6,
    'LastName': 'Suyama',
    'FirstName': 'Michael',
    'Title': 'Sales Representative',
    'TitleOfCourtesy': 'Mr.',
    'BirthDate': new Date(-205185600000),
    'HireDate': new Date(750830400000),
    'Address': 'Coventry House\r\nMiner Rd.',
    'City': 'London',
    'Region': null,
    'PostalCode': 'EC2 7JR',
    'Country': 'UK',
    'HomePhone': '(71) 555-7773',
    'Extension': '428',
    'Photo': { 'Length': 21626 },

```

```

'Notes': 'Michael is a graduate of Sussex University (MA, economics,
1983) and the University of California at Los Angeles \
(MBA, marketing, 1986). He has also taken the courses \'Multi-Cultural
Selling\' and \'Time Management for the Sales Professional. \' \
He is fluent in Japanese and can read and write French, Portuguese, and
Spanish.',
'ReportsTo': 5,
'PhotoPath': 'http://accweb/emmployees/davolio.bmp'
},
{
'EmployeeID': 7,
'LastName': 'King',
'FirstName': 'Robert',
'Title': 'Sales Representative',
'TitleOfCourtesy': 'Mr.',
'BirthDate': new Date(-302731200000),
'HireDate': new Date(757486800000),
'Address': 'Edgeham Hollow\r\nWinchester Way',
'City': 'London',
'Region': null,
'PostalCode': 'RG1 9SP',
'Country': 'UK',
'HomePhone': '(71) 555-5598',
'Extension': '465',
'Photo': { 'Length': 21626 },
'Notes': 'Robert King served in the Peace Corps and traveled extensively
before completing his degree in English at the \
University of Michigan in 1992, the year he joined the company. After
completing a course entitled \'Selling in Europe, \' \
he was transferred to the London office in March 1993.',
'ReportsTo': 5,
'PhotoPath': 'http://accweb/emmployees/davolio.bmp'
},
{
'EmployeeID': 8,
'LastName': 'Callahan',
'FirstName': 'Laura',
'Title': 'Inside Sales Coordinator',
'TitleOfCourtesy': 'Ms.',
'BirthDate': new Date(-377982000000),
'HireDate': new Date(762843600000),
'Address': '4726 - 11th Ave. N.E.',
'City': 'Seattle',
'Region': 'WA',
'PostalCode': '98105',
'Country': 'USA',
'HomePhone': '(206) 555-1189',
'Extension': '2344',
'Photo': { 'Length': 21626 },
'Notes': 'Laura received a BA in psychology from the University of
Washington. She has also completed a course in business \
French. She reads and writes French.',
'ReportsTo': 2,
'PhotoPath': 'http://accweb/emmployees/davolio.bmp'
},
{
'EmployeeID': 9,

```

```

        'LastName': 'Dodsworth',
        'FirstName': 'Anne',
        'Title': 'Sales Representative',
        'TitleOfCourtesy': 'Ms.',
        'BirthDate': new Date(-123966000000),
        'HireDate': new Date(784875600000),
        'Address': '7 Houndstooth Rd.',
        'City': 'London',
        'Region': null,
        'PostalCode': 'WG2 7LT',
        'Country': 'UK',
        'HomePhone': '(71) 555-4444',
        'Extension': '452',
        'Photo': { 'Length': 21626 },
        'Notes': 'Anne has a BA degree in English from St. Lawrence College. She is fluent in French and German.',
        'ReportsTo': 5,
        'PhotoPath': 'http://accweb/emmployees/davolio.bmp'
    }];
let stringData: string = JSON.stringify([
    {
        'OrderID': 10248,
        'CustomerID': 'VINET',
        'OrderDate': '1996-07-04T00:00:00.000Z',
        'ShippedDate': '1996-07-16T00:00:00.000Z',
        'Freight': 32.38,
        'ShipName': 'Vins et alcools Chevalier',
        'ShipAddress': '59 rue de l\'Abbaye',
        'ShipCity': 'Reims',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10249,
        'CustomerID': 'TOMSP',
        'OrderDate': '1996-07-05T00:00:00.000Z',
        'ShippedDate': '1996-07-10T00:00:00.000Z',
        'Freight': 11.61,
        'ShipName': 'Toms Spezialitäten',
        'ShipAddress': 'Luisenstr. 48',
        'ShipCity': 'Münster',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10250,
        'CustomerID': 'HANAR',
        'OrderDate': '1996-07-08T00:00:00.000Z',
        'ShippedDate': '1996-07-12T00:00:00.000Z',
        'Freight': 65.83,
        'ShipName': 'Hanari Carnes',
        'ShipAddress': 'Rua do Paço, 67',
        'ShipCity': 'Rio de Janeiro',
        'ShipRegion': 'RJ',
        'ShipCountry': 'Brazil'
    },
    {

```

```

    'OrderID': 10251,
    'CustomerID': 'VICTE',
    'OrderDate': '1996-07-08T00:00:00.000Z',
    'ShippedDate': '1996-07-15T00:00:00.000Z',
    'Freight': 41.34,
    'ShipName': 'Victuailles en stock',
    'ShipAddress': '2, rue du Commerce',
    'ShipCity': 'Lyon',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10252,
    'CustomerID': 'SUPRD',
    'OrderDate': '1996-07-09T00:00:00.000Z',
    'ShippedDate': '1996-07-11T00:00:00.000Z',
    'Freight': 51.3,
    'ShipName': 'Suprêmes délices',
    'ShipAddress': 'Boulevard Tirou, 255',
    'ShipCity': 'Charleroi',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10253,
    'CustomerID': 'HANAR',
    'OrderDate': '1996-07-10T00:00:00.000Z',
    'ShippedDate': '1996-07-16T00:00:00.000Z',
    'Freight': 58.17,
    'ShipName': 'Hanari Carnes',
    'ShipAddress': 'Rua do Paço, 67',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10254,
    'CustomerID': 'CHOPS',
    'OrderDate': '1996-07-11T00:00:00.000Z',
    'ShippedDate': '1996-07-23T00:00:00.000Z',
    'Freight': 22.98,
    'ShipName': 'Chop-suey Chinese',
    'ShipAddress': 'Hauptstr. 31',
    'ShipCity': 'Bern',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10255,
    'CustomerID': 'RICSU',
    'OrderDate': '1996-07-12T00:00:00.000Z',
    'ShippedDate': '1996-07-15T00:00:00.000Z',
    'Freight': 148.33,
    'ShipName': 'Richter Supermarkt',
    'ShipAddress': 'Starenweg 5',
    'ShipCity': 'Genève',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10256,
    'CustomerID': 'WELLI',
    'OrderDate': '1996-07-15T00:00:00.000Z',
    'ShippedDate': '1996-07-17T00:00:00.000Z',
    'Freight': 13.97,
    'ShipName': 'Wellington Importadora',
    'ShipAddress': 'Rua do Mercado, 12',
    'ShipCity': 'Resende',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10257,
    'CustomerID': 'HILAA',
    'OrderDate': '1996-07-16T00:00:00.000Z',
    'ShippedDate': '1996-07-22T00:00:00.000Z',
    'Freight': 81.91,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10258,
    'CustomerID': 'ERNSH',
    'OrderDate': '1996-07-17T00:00:00.000Z',
    'ShippedDate': '1996-07-23T00:00:00.000Z',
    'Freight': 140.51,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10259,
    'CustomerID': 'CENTC',
    'OrderDate': '1996-07-18T00:00:00.000Z',
    'ShippedDate': '1996-07-25T00:00:00.000Z',
    'Freight': 3.25,
    'ShipName': 'Centro comercial Moctezuma',
    'ShipAddress': 'Sierras de Granada 9993',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10260,
    'CustomerID': 'OTTIK',
    'OrderDate': '1996-07-19T00:00:00.000Z',
    'ShippedDate': '1996-07-29T00:00:00.000Z',
    'Freight': 55.09,
    'ShipName': 'Ottilies Käseladen',

```



```

    'ShipAddress': 'Mehrheimerstr. 369',
    'ShipCity': 'Köln',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10261,
    'CustomerID': 'QUEDE',
    'OrderDate': '1996-07-19T00:00:00.000Z',
    'ShippedDate': '1996-07-30T00:00:00.000Z',
    'Freight': 3.05,
    'ShipName': 'Que Delícia',
    'ShipAddress': 'Rua da Panificadora, 12',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10262,
    'CustomerID': 'RATTC',
    'OrderDate': '1996-07-22T00:00:00.000Z',
    'ShippedDate': '1996-07-25T00:00:00.000Z',
    'Freight': 48.29,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10263,
    'CustomerID': 'ERNSH',
    'OrderDate': '1996-07-23T00:00:00.000Z',
    'ShippedDate': '1996-07-31T00:00:00.000Z',
    'Freight': 146.06,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10264,
    'CustomerID': 'FOLKO',
    'OrderDate': '1996-07-24T00:00:00.000Z',
    'ShippedDate': '1996-08-23T00:00:00.000Z',
    'Freight': 3.67,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10265,
    'CustomerID': 'BLONP',
    'OrderDate': '1996-07-25T00:00:00.000Z',

```

```

    'ShippedDate': '1996-08-12T00:00:00.000Z',
    'Freight': 55.28,
    'ShipName': 'Blondel père et fils',
    'ShipAddress': '24, place Kléber',
    'ShipCity': 'Strasbourg',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10266,
    'CustomerID': 'WARTH',
    'OrderDate': '1996-07-26T00:00:00.000Z',
    'ShippedDate': '1996-07-31T00:00:00.000Z',
    'Freight': 25.73,
    'ShipName': 'Wartian Herkku',
    'ShipAddress': 'Torikatu 38',
    'ShipCity': 'Oulu',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10267,
    'CustomerID': 'FRANK',
    'OrderDate': '1996-07-29T00:00:00.000Z',
    'ShippedDate': '1996-08-06T00:00:00.000Z',
    'Freight': 208.58,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10268,
    'CustomerID': 'GROSR',
    'OrderDate': '1996-07-30T00:00:00.000Z',
    'ShippedDate': '1996-08-02T00:00:00.000Z',
    'Freight': 66.29,
    'ShipName': 'GROSELLA-Restaurante',
    'ShipAddress': '5a Ave. Los Palos Grandes',
    'ShipCity': 'Caracas',
    'ShipRegion': 'DF',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10269,
    'CustomerID': 'WHITC',
    'OrderDate': '1996-07-31T00:00:00.000Z',
    'ShippedDate': '1996-08-09T00:00:00.000Z',
    'Freight': 4.56,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {

```

```

    'OrderID': 10270,
    'CustomerID': 'WARTH',
    'OrderDate': '1996-08-01T00:00:00.000Z',
    'ShippedDate': '1996-08-02T00:00:00.000Z',
    'Freight': 136.54,
    'ShipName': 'Wartian Herkku',
    'ShipAddress': 'Torikatu 38',
    'ShipCity': 'Oulu',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10271,
    'CustomerID': 'SPLIR',
    'OrderDate': '1996-08-01T00:00:00.000Z',
    'ShippedDate': '1996-08-30T00:00:00.000Z',
    'Freight': 4.54,
    'ShipName': 'Split Rail Beer & Ale',
    'ShipAddress': 'P.O. Box 555',
    'ShipCity': 'Lander',
    'ShipRegion': 'WY',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10272,
    'CustomerID': 'RATTC',
    'OrderDate': '1996-08-02T00:00:00.000Z',
    'ShippedDate': '1996-08-06T00:00:00.000Z',
    'Freight': 98.03,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10273,
    'CustomerID': 'QUICK',
    'OrderDate': '1996-08-05T00:00:00.000Z',
    'ShippedDate': '1996-08-12T00:00:00.000Z',
    'Freight': 76.07,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10274,
    'CustomerID': 'VINET',
    'OrderDate': '1996-08-06T00:00:00.000Z',
    'ShippedDate': '1996-08-16T00:00:00.000Z',
    'Freight': 6.01,
    'ShipName': 'Vins et alcools Chevalier',
    'ShipAddress': '59 rue de l\'Abbaye',
    'ShipCity': 'Reims',
    'ShipRegion': null,

```

```

    'ShipCountry': 'France'
  },
  {
    'OrderID': 10275,
    'CustomerID': 'MAGAA',
    'OrderDate': '1996-08-07T00:00:00.000Z',
    'ShippedDate': '1996-08-09T00:00:00.000Z',
    'Freight': 26.93,
    'ShipName': 'Magazzini Alimentari Riuniti',
    'ShipAddress': 'Via Ludovico il Moro 22',
    'ShipCity': 'Bergamo',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10276,
    'CustomerID': 'TORTU',
    'OrderDate': '1996-08-08T00:00:00.000Z',
    'ShippedDate': '1996-08-14T00:00:00.000Z',
    'Freight': 13.84,
    'ShipName': 'Tortuga Restaurante',
    'ShipAddress': 'Avda. Azteca 123',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10277,
    'CustomerID': 'MORGK',
    'OrderDate': '1996-08-09T00:00:00.000Z',
    'ShippedDate': '1996-08-13T00:00:00.000Z',
    'Freight': 125.77,
    'ShipName': 'Morgenstern Gesundkost',
    'ShipAddress': 'Heerstr. 22',
    'ShipCity': 'Leipzig',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10278,
    'CustomerID': 'BERGS',
    'OrderDate': '1996-08-12T00:00:00.000Z',
    'ShippedDate': '1996-08-16T00:00:00.000Z',
    'Freight': 92.69,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10279,
    'CustomerID': 'LEHMS',
    'OrderDate': '1996-08-13T00:00:00.000Z',
    'ShippedDate': '1996-08-16T00:00:00.000Z',
    'Freight': 25.83,
    'ShipName': 'Lehmanns Marktstand',

```

```

        'ShipAddress': 'Magazinweg 7',
        'ShipCity': 'Frankfurt a.M.',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10280,
        'CustomerID': 'BERGS',
        'OrderDate': '1996-08-14T00:00:00.000Z',
        'ShippedDate': '1996-09-12T00:00:00.000Z',
        'Freight': 8.98,
        'ShipName': 'Berglunds snabbköp',
        'ShipAddress': 'Berguvsvägen 8',
        'ShipCity': 'Luleå',
        'ShipRegion': null,
        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 10281,
        'CustomerID': 'ROMEY',
        'OrderDate': '1996-08-14T00:00:00.000Z',
        'ShippedDate': '1996-08-21T00:00:00.000Z',
        'Freight': 2.94,
        'ShipName': 'Romero y tomillo',
        'ShipAddress': 'Gran Vía, 1',
        'ShipCity': 'Madrid',
        'ShipRegion': null,
        'ShipCountry': 'Spain'
    },
    {
        'OrderID': 10282,
        'CustomerID': 'ROMEY',
        'OrderDate': '1996-08-15T00:00:00.000Z',
        'ShippedDate': '1996-08-21T00:00:00.000Z',
        'Freight': 12.69,
        'ShipName': 'Romero y tomillo',
        'ShipAddress': 'Gran Vía, 1',
        'ShipCity': 'Madrid',
        'ShipRegion': null,
        'ShipCountry': 'Spain'
    },
    {
        'OrderID': 10283,
        'CustomerID': 'LILAS',
        'OrderDate': '1996-08-16T00:00:00.000Z',
        'ShippedDate': '1996-08-23T00:00:00.000Z',
        'Freight': 84.81,
        'ShipName': 'LILA-Supermercado',
        'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
        'ShipCity': 'Barquisimeto',
        'ShipRegion': 'Lara',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10284,
        'CustomerID': 'LEHMS',
        'OrderDate': '1996-08-19T00:00:00.000Z',

```

```

    'ShippedDate': '1996-08-27T00:00:00.000Z',
    'Freight': 76.56,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10285,
    'CustomerID': 'QUICK',
    'OrderDate': '1996-08-20T00:00:00.000Z',
    'ShippedDate': '1996-08-26T00:00:00.000Z',
    'Freight': 76.83,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10286,
    'CustomerID': 'QUICK',
    'OrderDate': '1996-08-21T00:00:00.000Z',
    'ShippedDate': '1996-08-30T00:00:00.000Z',
    'Freight': 229.24,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10287,
    'CustomerID': 'RICAR',
    'OrderDate': '1996-08-22T00:00:00.000Z',
    'ShippedDate': '1996-08-28T00:00:00.000Z',
    'Freight': 12.76,
    'ShipName': 'Ricardo Adocicados',
    'ShipAddress': 'Av. Copacabana, 267',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10288,
    'CustomerID': 'REGGC',
    'OrderDate': '1996-08-23T00:00:00.000Z',
    'ShippedDate': '1996-09-03T00:00:00.000Z',
    'Freight': 7.45,
    'ShipName': 'Reggiani Caseifici',
    'ShipAddress': 'Strada Provinciale 124',
    'ShipCity': 'Reggio Emilia',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {

```

```

      'OrderID': 10289,
      'CustomerID': 'BSBEV',
      'OrderDate': '1996-08-26T00:00:00.000Z',
      'ShippedDate': '1996-08-28T00:00:00.000Z',
      'Freight': 22.77,
      'ShipName': 'B\' Beverages',
      'ShipAddress': 'Fauntleroy Circus',
      'ShipCity': 'London',
      'ShipRegion': null,
      'ShipCountry': 'UK'
    },
    {
      'OrderID': 10290,
      'CustomerID': 'COMMI',
      'OrderDate': '1996-08-27T00:00:00.000Z',
      'ShippedDate': '1996-09-03T00:00:00.000Z',
      'Freight': 79.7,
      'ShipName': 'Comércio Mineiro',
      'ShipAddress': 'Av. dos Lusíadas, 23',
      'ShipCity': 'Sao Paulo',
      'ShipRegion': 'SP',
      'ShipCountry': 'Brazil'
    },
    {
      'OrderID': 10291,
      'CustomerID': 'QUEDE',
      'OrderDate': '1996-08-27T00:00:00.000Z',
      'ShippedDate': '1996-09-04T00:00:00.000Z',
      'Freight': 6.4,
      'ShipName': 'Que Delícia',
      'ShipAddress': 'Rua da Panificadora, 12',
      'ShipCity': 'Rio de Janeiro',
      'ShipRegion': 'RJ',
      'ShipCountry': 'Brazil'
    },
    {
      'OrderID': 10292,
      'CustomerID': 'TRADH',
      'OrderDate': '1996-08-28T00:00:00.000Z',
      'ShippedDate': '1996-09-02T00:00:00.000Z',
      'Freight': 1.35,
      'ShipName': 'Tradição Hipermercados',
      'ShipAddress': 'Av. Inês de Castro, 414',
      'ShipCity': 'Sao Paulo',
      'ShipRegion': 'SP',
      'ShipCountry': 'Brazil'
    },
    {
      'OrderID': 10293,
      'CustomerID': 'TORTU',
      'OrderDate': '1996-08-29T00:00:00.000Z',
      'ShippedDate': '1996-09-11T00:00:00.000Z',
      'Freight': 21.18,
      'ShipName': 'Tortuga Restaurante',
      'ShipAddress': 'Avda. Azteca 123',
      'ShipCity': 'México D.F.',
      'ShipRegion': null,

```

```

    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10294,
    'CustomerID': 'RATTC',
    'OrderDate': '1996-08-30T00:00:00.000Z',
    'ShippedDate': '1996-09-05T00:00:00.000Z',
    'Freight': 147.26,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10295,
    'CustomerID': 'VINET',
    'OrderDate': '1996-09-02T00:00:00.000Z',
    'ShippedDate': '1996-09-10T00:00:00.000Z',
    'Freight': 1.15,
    'ShipName': 'Vins et alcools Chevalier',
    'ShipAddress': '59 rue de l\'Abbaye',
    'ShipCity': 'Reims',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10296,
    'CustomerID': 'LILAS',
    'OrderDate': '1996-09-03T00:00:00.000Z',
    'ShippedDate': '1996-09-11T00:00:00.000Z',
    'Freight': 0.12,
    'ShipName': 'LILA-Supermercado',
    'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10297,
    'CustomerID': 'BLONP',
    'OrderDate': '1996-09-04T00:00:00.000Z',
    'ShippedDate': '1996-09-10T00:00:00.000Z',
    'Freight': 5.74,
    'ShipName': 'Blondel père et fils',
    'ShipAddress': '24, place Kléber',
    'ShipCity': 'Strasbourg',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10298,
    'CustomerID': 'HUNGO',
    'OrderDate': '1996-09-05T00:00:00.000Z',
    'ShippedDate': '1996-09-11T00:00:00.000Z',
    'Freight': 168.22,
    'ShipName': 'Hungry Owl All-Night Grocers',

```



```

        'ShipAddress': '8 Johnstown Road',
        'ShipCity': 'Cork',
        'ShipRegion': 'Co. Cork',
        'ShipCountry': 'Ireland'
    },
    {
        'OrderID': 10299,
        'CustomerID': 'RICAR',
        'OrderDate': '1996-09-06T00:00:00.000Z',
        'ShippedDate': '1996-09-13T00:00:00.000Z',
        'Freight': 29.76,
        'ShipName': 'Ricardo Adocicados',
        'ShipAddress': 'Av. Copacabana, 267',
        'ShipCity': 'Rio de Janeiro',
        'ShipRegion': 'RJ',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10300,
        'CustomerID': 'MAGAA',
        'OrderDate': '1996-09-09T00:00:00.000Z',
        'ShippedDate': '1996-09-18T00:00:00.000Z',
        'Freight': 17.68,
        'ShipName': 'Magazzini Alimentari Riuniti',
        'ShipAddress': 'Via Ludovico il Moro 22',
        'ShipCity': 'Bergamo',
        'ShipRegion': null,
        'ShipCountry': 'Italy'
    },
    {
        'OrderID': 10301,
        'CustomerID': 'WANDK',
        'OrderDate': '1996-09-09T00:00:00.000Z',
        'ShippedDate': '1996-09-17T00:00:00.000Z',
        'Freight': 45.08,
        'ShipName': 'Die Wandernde Kuh',
        'ShipAddress': 'Adenauerallee 900',
        'ShipCity': 'Stuttgart',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10302,
        'CustomerID': 'SUPRD',
        'OrderDate': '1996-09-10T00:00:00.000Z',
        'ShippedDate': '1996-10-09T00:00:00.000Z',
        'Freight': 6.27,
        'ShipName': 'Suprêmes délices',
        'ShipAddress': 'Boulevard Tirou, 255',
        'ShipCity': 'Charleroi',
        'ShipRegion': null,
        'ShipCountry': 'Belgium'
    },
    {
        'OrderID': 10303,
        'CustomerID': 'GODOS',
        'OrderDate': '1996-09-11T00:00:00.000Z',

```

```

        'ShippedDate': '1996-09-18T00:00:00.000Z',
        'Freight': 107.83,
        'ShipName': 'Godos Cocina Típica',
        'ShipAddress': 'C/ Romero, 33',
        'ShipCity': 'Sevilla',
        'ShipRegion': null,
        'ShipCountry': 'Spain'
    },
    {
        'OrderID': 10304,
        'CustomerID': 'TORTU',
        'OrderDate': '1996-09-12T00:00:00.000Z',
        'ShippedDate': '1996-09-17T00:00:00.000Z',
        'Freight': 63.79,
        'ShipName': 'Tortuga Restaurante',
        'ShipAddress': 'Avda. Azteca 123',
        'ShipCity': 'México D.F.',
        'ShipRegion': null,
        'ShipCountry': 'Mexico'
    },
    {
        'OrderID': 10305,
        'CustomerID': 'OLDWO',
        'OrderDate': '1996-09-13T00:00:00.000Z',
        'ShippedDate': '1996-10-09T00:00:00.000Z',
        'Freight': 257.62,
        'ShipName': 'Old World Delicatessen',
        'ShipAddress': '2743 Bering St.',
        'ShipCity': 'Anchorage',
        'ShipRegion': 'AK',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10306,
        'CustomerID': 'ROMEY',
        'OrderDate': '1996-09-16T00:00:00.000Z',
        'ShippedDate': '1996-09-23T00:00:00.000Z',
        'Freight': 7.56,
        'ShipName': 'Romero y tomillo',
        'ShipAddress': 'Gran Vía, 1',
        'ShipCity': 'Madrid',
        'ShipRegion': null,
        'ShipCountry': 'Spain'
    },
    {
        'OrderID': 10307,
        'CustomerID': 'LONEP',
        'OrderDate': '1996-09-17T00:00:00.000Z',
        'ShippedDate': '1996-09-25T00:00:00.000Z',
        'Freight': 0.56,
        'ShipName': 'Lonesome Pine Restaurant',
        'ShipAddress': '89 Chiaroscuro Rd.',
        'ShipCity': 'Portland',
        'ShipRegion': 'OR',
        'ShipCountry': 'USA'
    }
]

```

```

    'OrderID': 10308,
    'CustomerID': 'ANATR',
    'OrderDate': '1996-09-18T00:00:00.000Z',
    'ShippedDate': '1996-09-24T00:00:00.000Z',
    'Freight': 1.61,
    'ShipName': 'Ana Trujillo Emparedados y helados',
    'ShipAddress': 'Avda. de la Constitución 2222',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10309,
    'CustomerID': 'HUNGO',
    'OrderDate': '1996-09-19T00:00:00.000Z',
    'ShippedDate': '1996-10-23T00:00:00.000Z',
    'Freight': 47.3,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10310,
    'CustomerID': 'THEBI',
    'OrderDate': '1996-09-20T00:00:00.000Z',
    'ShippedDate': '1996-09-27T00:00:00.000Z',
    'Freight': 17.52,
    'ShipName': 'The Big Cheese',
    'ShipAddress': '89 Jefferson Way Suite 2',
    'ShipCity': 'Portland',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10311,
    'CustomerID': 'DUMON',
    'OrderDate': '1996-09-20T00:00:00.000Z',
    'ShippedDate': '1996-09-26T00:00:00.000Z',
    'Freight': 24.69,
    'ShipName': 'Du monde entier',
    'ShipAddress': '67, rue des Cinquante Otages',
    'ShipCity': 'Nantes',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10312,
    'CustomerID': 'WANDK',
    'OrderDate': '1996-09-23T00:00:00.000Z',
    'ShippedDate': '1996-10-03T00:00:00.000Z',
    'Freight': 40.26,
    'ShipName': 'Die Wandernde Kuh',
    'ShipAddress': 'Adenauerallee 900',
    'ShipCity': 'Stuttgart',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10313,
    'CustomerID': 'QUICK',
    'OrderDate': '1996-09-24T00:00:00.000Z',
    'ShippedDate': '1996-10-04T00:00:00.000Z',
    'Freight': 1.96,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10314,
    'CustomerID': 'RATTC',
    'OrderDate': '1996-09-25T00:00:00.000Z',
    'ShippedDate': '1996-10-04T00:00:00.000Z',
    'Freight': 74.16,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10315,
    'CustomerID': 'ISLAT',
    'OrderDate': '1996-09-26T00:00:00.000Z',
    'ShippedDate': '1996-10-03T00:00:00.000Z',
    'Freight': 41.76,
    'ShipName': 'Island Trading',
    'ShipAddress': 'Garden House Crowther Way',
    'ShipCity': 'Cowes',
    'ShipRegion': 'Isle of Wight',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10316,
    'CustomerID': 'RATTC',
    'OrderDate': '1996-09-27T00:00:00.000Z',
    'ShippedDate': '1996-10-08T00:00:00.000Z',
    'Freight': 150.15,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10317,
    'CustomerID': 'LONEP',
    'OrderDate': '1996-09-30T00:00:00.000Z',
    'ShippedDate': '1996-10-10T00:00:00.000Z',
    'Freight': 12.69,
    'ShipName': 'Lonesome Pine Restaurant',

```

```

        'ShipAddress': '89 Chiaroscuro Rd.',
        'ShipCity': 'Portland',
        'ShipRegion': 'OR',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10318,
        'CustomerID': 'ISLAT',
        'OrderDate': '1996-10-01T00:00:00.000Z',
        'ShippedDate': '1996-10-04T00:00:00.000Z',
        'Freight': 4.73,
        'ShipName': 'Island Trading',
        'ShipAddress': 'Garden House Crowther Way',
        'ShipCity': 'Cowes',
        'ShipRegion': 'Isle of Wight',
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10319,
        'CustomerID': 'TORTU',
        'OrderDate': '1996-10-02T00:00:00.000Z',
        'ShippedDate': '1996-10-11T00:00:00.000Z',
        'Freight': 64.5,
        'ShipName': 'Tortuga Restaurante',
        'ShipAddress': 'Avda. Azteca 123',
        'ShipCity': 'México D.F.',
        'ShipRegion': null,
        'ShipCountry': 'Mexico'
    },
    {
        'OrderID': 10320,
        'CustomerID': 'WARTH',
        'OrderDate': '1996-10-03T00:00:00.000Z',
        'ShippedDate': '1996-10-18T00:00:00.000Z',
        'Freight': 34.57,
        'ShipName': 'Wartian Herkku',
        'ShipAddress': 'Torikatu 38',
        'ShipCity': 'Oulu',
        'ShipRegion': null,
        'ShipCountry': 'Finland'
    },
    {
        'OrderID': 10321,
        'CustomerID': 'ISLAT',
        'OrderDate': '1996-10-03T00:00:00.000Z',
        'ShippedDate': '1996-10-11T00:00:00.000Z',
        'Freight': 3.43,
        'ShipName': 'Island Trading',
        'ShipAddress': 'Garden House Crowther Way',
        'ShipCity': 'Cowes',
        'ShipRegion': 'Isle of Wight',
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10322,
        'CustomerID': 'PERIC',
        'OrderDate': '1996-10-04T00:00:00.000Z',

```

```

        'ShippedDate': '1996-10-23T00:00:00.000Z',
        'Freight': 0.4,
        'ShipName': 'Pericles Comidas clásicas',
        'ShipAddress': 'Calle Dr. Jorge Cash 321',
        'ShipCity': 'México D.F.',
        'ShipRegion': null,
        'ShipCountry': 'Mexico'
    },
    {
        'OrderID': 10323,
        'CustomerID': 'KOENE',
        'OrderDate': '1996-10-07T00:00:00.000Z',
        'ShippedDate': '1996-10-14T00:00:00.000Z',
        'Freight': 4.88,
        'ShipName': 'Königlich Essen',
        'ShipAddress': 'Maubelstr. 90',
        'ShipCity': 'Brandenburg',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10324,
        'CustomerID': 'SAVEA',
        'OrderDate': '1996-10-08T00:00:00.000Z',
        'ShippedDate': '1996-10-10T00:00:00.000Z',
        'Freight': 214.27,
        'ShipName': 'Save-a-lot Markets',
        'ShipAddress': '187 Suffolk Ln.',
        'ShipCity': 'Boise',
        'ShipRegion': 'ID',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10325,
        'CustomerID': 'KOENE',
        'OrderDate': '1996-10-09T00:00:00.000Z',
        'ShippedDate': '1996-10-14T00:00:00.000Z',
        'Freight': 64.86,
        'ShipName': 'Königlich Essen',
        'ShipAddress': 'Maubelstr. 90',
        'ShipCity': 'Brandenburg',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10326,
        'CustomerID': 'BOLID',
        'OrderDate': '1996-10-10T00:00:00.000Z',
        'ShippedDate': '1996-10-14T00:00:00.000Z',
        'Freight': 77.92,
        'ShipName': 'Bólido Comidas preparadas',
        'ShipAddress': 'C/ Araquil, 67',
        'ShipCity': 'Madrid',
        'ShipRegion': null,
        'ShipCountry': 'Spain'
    },
    {

```

```

    'OrderID': 10327,
    'CustomerID': 'FOLKO',
    'OrderDate': '1996-10-11T00:00:00.000Z',
    'ShippedDate': '1996-10-14T00:00:00.000Z',
    'Freight': 63.36,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10328,
    'CustomerID': 'FURIB',
    'OrderDate': '1996-10-14T00:00:00.000Z',
    'ShippedDate': '1996-10-17T00:00:00.000Z',
    'Freight': 87.03,
    'ShipName': 'Furia Bacalhau e Frutos do Mar',
    'ShipAddress': 'Jardim das rosas n. 32',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,
    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 10329,
    'CustomerID': 'SPLIR',
    'OrderDate': '1996-10-15T00:00:00.000Z',
    'ShippedDate': '1996-10-23T00:00:00.000Z',
    'Freight': 191.67,
    'ShipName': 'Split Rail Beer & Ale',
    'ShipAddress': 'P.O. Box 555',
    'ShipCity': 'Lander',
    'ShipRegion': 'WY',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10330,
    'CustomerID': 'LILAS',
    'OrderDate': '1996-10-16T00:00:00.000Z',
    'ShippedDate': '1996-10-28T00:00:00.000Z',
    'Freight': 12.75,
    'ShipName': 'LILA-Supermercado',
    'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10331,
    'CustomerID': 'BONAP',
    'OrderDate': '1996-10-16T00:00:00.000Z',
    'ShippedDate': '1996-10-21T00:00:00.000Z',
    'Freight': 10.19,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,

```

```

    'ShipCountry': 'France'
  },
  {
    'OrderID': 10332,
    'CustomerID': 'MEREP',
    'OrderDate': '1996-10-17T00:00:00.000Z',
    'ShippedDate': '1996-10-21T00:00:00.000Z',
    'Freight': 52.84,
    'ShipName': 'Mère Paillarde',
    'ShipAddress': '43 rue St. Laurent',
    'ShipCity': 'Montréal',
    'ShipRegion': 'Québec',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10333,
    'CustomerID': 'WARTH',
    'OrderDate': '1996-10-18T00:00:00.000Z',
    'ShippedDate': '1996-10-25T00:00:00.000Z',
    'Freight': 0.59,
    'ShipName': 'Wartian Herkku',
    'ShipAddress': 'Torikatu 38',
    'ShipCity': 'Oulu',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10334,
    'CustomerID': 'VICTE',
    'OrderDate': '1996-10-21T00:00:00.000Z',
    'ShippedDate': '1996-10-28T00:00:00.000Z',
    'Freight': 8.56,
    'ShipName': 'Victuailles en stock',
    'ShipAddress': '2, rue du Commerce',
    'ShipCity': 'Lyon',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10335,
    'CustomerID': 'HUNGO',
    'OrderDate': '1996-10-22T00:00:00.000Z',
    'ShippedDate': '1996-10-24T00:00:00.000Z',
    'Freight': 42.11,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10336,
    'CustomerID': 'PRINI',
    'OrderDate': '1996-10-23T00:00:00.000Z',
    'ShippedDate': '1996-10-25T00:00:00.000Z',
    'Freight': 15.51,
    'ShipName': 'Princesa Isabel Vinhos',

```



```

    'ShipAddress': 'Estrada da saúde n. 58',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,
    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 10337,
    'CustomerID': 'FRANK',
    'OrderDate': '1996-10-24T00:00:00.000Z',
    'ShippedDate': '1996-10-29T00:00:00.000Z',
    'Freight': 108.26,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10338,
    'CustomerID': 'OLDWO',
    'OrderDate': '1996-10-25T00:00:00.000Z',
    'ShippedDate': '1996-10-29T00:00:00.000Z',
    'Freight': 84.21,
    'ShipName': 'Old World Delicatessen',
    'ShipAddress': '2743 Bering St.',
    'ShipCity': 'Anchorage',
    'ShipRegion': 'AK',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10339,
    'CustomerID': 'MEREP',
    'OrderDate': '1996-10-28T00:00:00.000Z',
    'ShippedDate': '1996-11-04T00:00:00.000Z',
    'Freight': 15.66,
    'ShipName': 'Mère Paillarde',
    'ShipAddress': '43 rue St. Laurent',
    'ShipCity': 'Montréal',
    'ShipRegion': 'Québec',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10340,
    'CustomerID': 'BONAP',
    'OrderDate': '1996-10-29T00:00:00.000Z',
    'ShippedDate': '1996-11-08T00:00:00.000Z',
    'Freight': 166.31,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10341,
    'CustomerID': 'SIMOB',
    'OrderDate': '1996-10-29T00:00:00.000Z',

```

```

    'ShippedDate': '1996-11-05T00:00:00.000Z',
    'Freight': 26.78,
    'ShipName': 'Simons bistro',
    'ShipAddress': 'Vinbæltet 34',
    'ShipCity': 'Kobenhavn',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10342,
    'CustomerID': 'FRANK',
    'OrderDate': '1996-10-30T00:00:00.000Z',
    'ShippedDate': '1996-11-04T00:00:00.000Z',
    'Freight': 54.83,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10343,
    'CustomerID': 'LEHMS',
    'OrderDate': '1996-10-31T00:00:00.000Z',
    'ShippedDate': '1996-11-06T00:00:00.000Z',
    'Freight': 110.37,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10344,
    'CustomerID': 'WHITC',
    'OrderDate': '1996-11-01T00:00:00.000Z',
    'ShippedDate': '1996-11-05T00:00:00.000Z',
    'Freight': 23.29,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10345,
    'CustomerID': 'QUICK',
    'OrderDate': '1996-11-04T00:00:00.000Z',
    'ShippedDate': '1996-11-11T00:00:00.000Z',
    'Freight': 249.06,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {

```

```

      'OrderID': 10346,
      'CustomerID': 'RATTC',
      'OrderDate': '1996-11-05T00:00:00.000Z',
      'ShippedDate': '1996-11-08T00:00:00.000Z',
      'Freight': 142.08,
      'ShipName': 'Rattlesnake Canyon Grocery',
      'ShipAddress': '2817 Milton Dr.',
      'ShipCity': 'Albuquerque',
      'ShipRegion': 'NM',
      'ShipCountry': 'USA'
    },
    {
      'OrderID': 10347,
      'CustomerID': 'FAMIA',
      'OrderDate': '1996-11-06T00:00:00.000Z',
      'ShippedDate': '1996-11-08T00:00:00.000Z',
      'Freight': 3.1,
      'ShipName': 'Familia Arquibaldo',
      'ShipAddress': 'Rua Orós, 92',
      'ShipCity': 'Sao Paulo',
      'ShipRegion': 'SP',
      'ShipCountry': 'Brazil'
    },
    {
      'OrderID': 10348,
      'CustomerID': 'WANDK',
      'OrderDate': '1996-11-07T00:00:00.000Z',
      'ShippedDate': '1996-11-15T00:00:00.000Z',
      'Freight': 0.78,
      'ShipName': 'Die Wandernde Kuh',
      'ShipAddress': 'Adenauerallee 900',
      'ShipCity': 'Stuttgart',
      'ShipRegion': null,
      'ShipCountry': 'Germany'
    },
    {
      'OrderID': 10349,
      'CustomerID': 'SPLIR',
      'OrderDate': '1996-11-08T00:00:00.000Z',
      'ShippedDate': '1996-11-15T00:00:00.000Z',
      'Freight': 8.63,
      'ShipName': 'Split Rail Beer & Ale',
      'ShipAddress': 'P.O. Box 555',
      'ShipCity': 'Lander',
      'ShipRegion': 'WY',
      'ShipCountry': 'USA'
    },
    {
      'OrderID': 10350,
      'CustomerID': 'LAMAI',
      'OrderDate': '1996-11-11T00:00:00.000Z',
      'ShippedDate': '1996-12-03T00:00:00.000Z',
      'Freight': 64.19,
      'ShipName': 'La maison d\'Asie',
      'ShipAddress': '1 rue Alsace-Lorraine',
      'ShipCity': 'Toulouse',
      'ShipRegion': null,

```

```

    'ShipCountry': 'France'
  },
  {
    'OrderID': 10351,
    'CustomerID': 'ERNSH',
    'OrderDate': '1996-11-11T00:00:00.000Z',
    'ShippedDate': '1996-11-20T00:00:00.000Z',
    'Freight': 162.33,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10352,
    'CustomerID': 'FURIB',
    'OrderDate': '1996-11-12T00:00:00.000Z',
    'ShippedDate': '1996-11-18T00:00:00.000Z',
    'Freight': 1.3,
    'ShipName': 'Furia Bacalhau e Frutos do Mar',
    'ShipAddress': 'Jardim das rosas n. 32',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,
    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 10353,
    'CustomerID': 'PICCO',
    'OrderDate': '1996-11-13T00:00:00.000Z',
    'ShippedDate': '1996-11-25T00:00:00.000Z',
    'Freight': 360.63,
    'ShipName': 'Piccolo und mehr',
    'ShipAddress': 'Geislweg 14',
    'ShipCity': 'Salzburg',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10354,
    'CustomerID': 'PERIC',
    'OrderDate': '1996-11-14T00:00:00.000Z',
    'ShippedDate': '1996-11-20T00:00:00.000Z',
    'Freight': 53.8,
    'ShipName': 'Pericles Comidas clásicas',
    'ShipAddress': 'Calle Dr. Jorge Cash 321',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10355,
    'CustomerID': 'AROUT',
    'OrderDate': '1996-11-15T00:00:00.000Z',
    'ShippedDate': '1996-11-20T00:00:00.000Z',
    'Freight': 41.95,
    'ShipName': 'Around the Horn',

```

```

        'ShipAddress': 'Brook Farm Stratford St. Mary',
        'ShipCity': 'Colchester',
        'ShipRegion': 'Essex',
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10356,
        'CustomerID': 'WANDK',
        'OrderDate': '1996-11-18T00:00:00.000Z',
        'ShippedDate': '1996-11-27T00:00:00.000Z',
        'Freight': 36.71,
        'ShipName': 'Die Wandernde Kuh',
        'ShipAddress': 'Adenauerallee 900',
        'ShipCity': 'Stuttgart',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10357,
        'CustomerID': 'LILAS',
        'OrderDate': '1996-11-19T00:00:00.000Z',
        'ShippedDate': '1996-12-02T00:00:00.000Z',
        'Freight': 34.88,
        'ShipName': 'LILA-Supermercado',
        'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
        'ShipCity': 'Barquisimeto',
        'ShipRegion': 'Lara',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10358,
        'CustomerID': 'LAMAI',
        'OrderDate': '1996-11-20T00:00:00.000Z',
        'ShippedDate': '1996-11-27T00:00:00.000Z',
        'Freight': 19.64,
        'ShipName': 'La maison d\'Asie',
        'ShipAddress': '1 rue Alsace-Lorraine',
        'ShipCity': 'Toulouse',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10359,
        'CustomerID': 'SEVES',
        'OrderDate': '1996-11-21T00:00:00.000Z',
        'ShippedDate': '1996-11-26T00:00:00.000Z',
        'Freight': 288.43,
        'ShipName': 'Seven Seas Imports',
        'ShipAddress': '90 Wadhurst Rd.',
        'ShipCity': 'London',
        'ShipRegion': null,
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10360,
        'CustomerID': 'BLONP',
        'OrderDate': '1996-11-22T00:00:00.000Z',

```

```

    'ShippedDate': '1996-12-02T00:00:00.000Z',
    'Freight': 131.7,
    'ShipName': 'Blondel père et fils',
    'ShipAddress': '24, place Kléber',
    'ShipCity': 'Strasbourg',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10361,
    'CustomerID': 'QUICK',
    'OrderDate': '1996-11-22T00:00:00.000Z',
    'ShippedDate': '1996-12-03T00:00:00.000Z',
    'Freight': 183.17,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10362,
    'CustomerID': 'BONAP',
    'OrderDate': '1996-11-25T00:00:00.000Z',
    'ShippedDate': '1996-11-28T00:00:00.000Z',
    'Freight': 96.04,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10363,
    'CustomerID': 'DRACD',
    'OrderDate': '1996-11-26T00:00:00.000Z',
    'ShippedDate': '1996-12-04T00:00:00.000Z',
    'Freight': 30.54,
    'ShipName': 'Drachenblut Delikatessen',
    'ShipAddress': 'Walserweg 21',
    'ShipCity': 'Aachen',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10364,
    'CustomerID': 'EASTC',
    'OrderDate': '1996-11-26T00:00:00.000Z',
    'ShippedDate': '1996-12-04T00:00:00.000Z',
    'Freight': 71.97,
    'ShipName': 'Eastern Connection',
    'ShipAddress': '35 King George',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {

```

```

    'OrderID': 10365,
    'CustomerID': 'ANTON',
    'OrderDate': '1996-11-27T00:00:00.000Z',
    'ShippedDate': '1996-12-02T00:00:00.000Z',
    'Freight': 22,
    'ShipName': 'Antonio Moreno Taquería',
    'ShipAddress': 'Mataderos 2312',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10366,
    'CustomerID': 'GALED',
    'OrderDate': '1996-11-28T00:00:00.000Z',
    'ShippedDate': '1996-12-30T00:00:00.000Z',
    'Freight': 10.14,
    'ShipName': 'Galería del gastronómo',
    'ShipAddress': 'Rambla de Cataluña, 23',
    'ShipCity': 'Barcelona',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10367,
    'CustomerID': 'VAFFE',
    'OrderDate': '1996-11-28T00:00:00.000Z',
    'ShippedDate': '1996-12-02T00:00:00.000Z',
    'Freight': 13.55,
    'ShipName': 'Vaffeljernet',
    'ShipAddress': 'Smagsloget 45',
    'ShipCity': 'Århus',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10368,
    'CustomerID': 'ERNSH',
    'OrderDate': '1996-11-29T00:00:00.000Z',
    'ShippedDate': '1996-12-02T00:00:00.000Z',
    'Freight': 101.95,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10369,
    'CustomerID': 'SPLIR',
    'OrderDate': '1996-12-02T00:00:00.000Z',
    'ShippedDate': '1996-12-09T00:00:00.000Z',
    'Freight': 195.68,
    'ShipName': 'Split Rail Beer & Ale',
    'ShipAddress': 'P.O. Box 555',
    'ShipCity': 'Lander',
    'ShipRegion': 'WY',

```

```

    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10370,
    'CustomerID': 'CHOPS',
    'OrderDate': '1996-12-03T00:00:00.000Z',
    'ShippedDate': '1996-12-27T00:00:00.000Z',
    'Freight': 1.17,
    'ShipName': 'Chop-suey Chinese',
    'ShipAddress': 'Hauptstr. 31',
    'ShipCity': 'Bern',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10371,
    'CustomerID': 'LAMAI',
    'OrderDate': '1996-12-03T00:00:00.000Z',
    'ShippedDate': '1996-12-24T00:00:00.000Z',
    'Freight': 0.45,
    'ShipName': 'La maison d\'Asie',
    'ShipAddress': '1 rue Alsace-Lorraine',
    'ShipCity': 'Toulouse',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10372,
    'CustomerID': 'QUEEN',
    'OrderDate': '1996-12-04T00:00:00.000Z',
    'ShippedDate': '1996-12-09T00:00:00.000Z',
    'Freight': 890.78,
    'ShipName': 'Queen Cozinha',
    'ShipAddress': 'Alameda dos Canários, 891',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10373,
    'CustomerID': 'HUNGO',
    'OrderDate': '1996-12-05T00:00:00.000Z',
    'ShippedDate': '1996-12-11T00:00:00.000Z',
    'Freight': 124.12,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10374,
    'CustomerID': 'WOLZA',
    'OrderDate': '1996-12-05T00:00:00.000Z',
    'ShippedDate': '1996-12-09T00:00:00.000Z',
    'Freight': 3.94,
    'ShipName': 'Wolski Zajazd',

```



```

        'ShipAddress': 'ul. Filtrowa 68',
        'ShipCity': 'Warszawa',
        'ShipRegion': null,
        'ShipCountry': 'Poland'
    },
    {
        'OrderID': 10375,
        'CustomerID': 'HUNGC',
        'OrderDate': '1996-12-06T00:00:00.000Z',
        'ShippedDate': '1996-12-09T00:00:00.000Z',
        'Freight': 20.12,
        'ShipName': 'Hungry Coyote Import Store',
        'ShipAddress': 'City Center Plaza 516 Main St.',
        'ShipCity': 'Elgin',
        'ShipRegion': 'OR',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10376,
        'CustomerID': 'MEREP',
        'OrderDate': '1996-12-09T00:00:00.000Z',
        'ShippedDate': '1996-12-13T00:00:00.000Z',
        'Freight': 20.39,
        'ShipName': 'Mère Paillarde',
        'ShipAddress': '43 rue St. Laurent',
        'ShipCity': 'Montréal',
        'ShipRegion': 'Québec',
        'ShipCountry': 'Canada'
    },
    {
        'OrderID': 10377,
        'CustomerID': 'SEVES',
        'OrderDate': '1996-12-09T00:00:00.000Z',
        'ShippedDate': '1996-12-13T00:00:00.000Z',
        'Freight': 22.21,
        'ShipName': 'Seven Seas Imports',
        'ShipAddress': '90 Wadhurst Rd.',
        'ShipCity': 'London',
        'ShipRegion': null,
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10378,
        'CustomerID': 'FOLKO',
        'OrderDate': '1996-12-10T00:00:00.000Z',
        'ShippedDate': '1996-12-19T00:00:00.000Z',
        'Freight': 5.44,
        'ShipName': 'Folk och fä HB',
        'ShipAddress': 'Åkergatan 24',
        'ShipCity': 'Bräcke',
        'ShipRegion': null,
        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 10379,
        'CustomerID': 'QUEDE',
        'OrderDate': '1996-12-11T00:00:00.000Z',

```

```

    'ShippedDate': '1996-12-13T00:00:00.000Z',
    'Freight': 45.03,
    'ShipName': 'Que Delícia',
    'ShipAddress': 'Rua da Panificadora, 12',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10380,
    'CustomerID': 'HUNGO',
    'OrderDate': '1996-12-12T00:00:00.000Z',
    'ShippedDate': '1997-01-16T00:00:00.000Z',
    'Freight': 35.03,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10381,
    'CustomerID': 'LILAS',
    'OrderDate': '1996-12-12T00:00:00.000Z',
    'ShippedDate': '1996-12-13T00:00:00.000Z',
    'Freight': 7.99,
    'ShipName': 'LILA-Supermercado',
    'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10382,
    'CustomerID': 'ERNSH',
    'OrderDate': '1996-12-13T00:00:00.000Z',
    'ShippedDate': '1996-12-16T00:00:00.000Z',
    'Freight': 94.77,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10383,
    'CustomerID': 'AROUT',
    'OrderDate': '1996-12-16T00:00:00.000Z',
    'ShippedDate': '1996-12-18T00:00:00.000Z',
    'Freight': 34.24,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',
    'ShipCountry': 'UK'
  }
]

```

```

    'OrderID': 10384,
    'CustomerID': 'BERGS',
    'OrderDate': '1996-12-16T00:00:00.000Z',
    'ShippedDate': '1996-12-20T00:00:00.000Z',
    'Freight': 168.64,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10385,
    'CustomerID': 'SPLIR',
    'OrderDate': '1996-12-17T00:00:00.000Z',
    'ShippedDate': '1996-12-23T00:00:00.000Z',
    'Freight': 30.96,
    'ShipName': 'Split Rail Beer & Ale',
    'ShipAddress': 'P.O. Box 555',
    'ShipCity': 'Lander',
    'ShipRegion': 'WY',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10386,
    'CustomerID': 'FAMIA',
    'OrderDate': '1996-12-18T00:00:00.000Z',
    'ShippedDate': '1996-12-25T00:00:00.000Z',
    'Freight': 13.99,
    'ShipName': 'Familia Arquibaldo',
    'ShipAddress': 'Rua Orós, 92',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10387,
    'CustomerID': 'SANTG',
    'OrderDate': '1996-12-18T00:00:00.000Z',
    'ShippedDate': '1996-12-20T00:00:00.000Z',
    'Freight': 93.63,
    'ShipName': 'Santé Gourmet',
    'ShipAddress': 'Erling Skakkes gate 78',
    'ShipCity': 'Stavern',
    'ShipRegion': null,
    'ShipCountry': 'Norway'
  },
  {
    'OrderID': 10388,
    'CustomerID': 'SEVES',
    'OrderDate': '1996-12-19T00:00:00.000Z',
    'ShippedDate': '1996-12-20T00:00:00.000Z',
    'Freight': 34.86,
    'ShipName': 'Seven Seas Imports',
    'ShipAddress': '90 Wadhurst Rd.',
    'ShipCity': 'London',
    'ShipRegion': null,

```

```

        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10389,
        'CustomerID': 'BOTTM',
        'OrderDate': '1996-12-20T00:00:00.000Z',
        'ShippedDate': '1996-12-24T00:00:00.000Z',
        'Freight': 47.42,
        'ShipName': 'Bottom-Dollar Markets',
        'ShipAddress': '23 Tsawassen Blvd.',
        'ShipCity': 'Tsawassen',
        'ShipRegion': 'BC',
        'ShipCountry': 'Canada'
    },
    {
        'OrderID': 10390,
        'CustomerID': 'ERNSH',
        'OrderDate': '1996-12-23T00:00:00.000Z',
        'ShippedDate': '1996-12-26T00:00:00.000Z',
        'Freight': 126.38,
        'ShipName': 'Ernst Handel',
        'ShipAddress': 'Kirchgasse 6',
        'ShipCity': 'Graz',
        'ShipRegion': null,
        'ShipCountry': 'Austria'
    },
    {
        'OrderID': 10391,
        'CustomerID': 'DRACD',
        'OrderDate': '1996-12-23T00:00:00.000Z',
        'ShippedDate': '1996-12-31T00:00:00.000Z',
        'Freight': 5.45,
        'ShipName': 'Drachenblut Delikatessen',
        'ShipAddress': 'Walserweg 21',
        'ShipCity': 'Aachen',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10392,
        'CustomerID': 'PICCO',
        'OrderDate': '1996-12-24T00:00:00.000Z',
        'ShippedDate': '1997-01-01T00:00:00.000Z',
        'Freight': 122.46,
        'ShipName': 'Piccolo und mehr',
        'ShipAddress': 'Geislweg 14',
        'ShipCity': 'Salzburg',
        'ShipRegion': null,
        'ShipCountry': 'Austria'
    },
    {
        'OrderID': 10393,
        'CustomerID': 'SAVEA',
        'OrderDate': '1996-12-25T00:00:00.000Z',
        'ShippedDate': '1997-01-03T00:00:00.000Z',
        'Freight': 126.56,
        'ShipName': 'Save-a-lot Markets',

```

```

        'ShipAddress': '187 Suffolk Ln.',
        'ShipCity': 'Boise',
        'ShipRegion': 'ID',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10394,
        'CustomerID': 'HUNGC',
        'OrderDate': '1996-12-25T00:00:00.000Z',
        'ShippedDate': '1997-01-03T00:00:00.000Z',
        'Freight': 30.34,
        'ShipName': 'Hungry Coyote Import Store',
        'ShipAddress': 'City Center Plaza 516 Main St.',
        'ShipCity': 'Elgin',
        'ShipRegion': 'OR',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10395,
        'CustomerID': 'HILAA',
        'OrderDate': '1996-12-26T00:00:00.000Z',
        'ShippedDate': '1997-01-03T00:00:00.000Z',
        'Freight': 184.41,
        'ShipName': 'HILARION-Abastos',
        'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
        'ShipCity': 'San Cristóbal',
        'ShipRegion': 'Táchira',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10396,
        'CustomerID': 'FRANK',
        'OrderDate': '1996-12-27T00:00:00.000Z',
        'ShippedDate': '1997-01-06T00:00:00.000Z',
        'Freight': 135.35,
        'ShipName': 'Frankenversand',
        'ShipAddress': 'Berliner Platz 43',
        'ShipCity': 'München',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10397,
        'CustomerID': 'PRINI',
        'OrderDate': '1996-12-27T00:00:00.000Z',
        'ShippedDate': '1997-01-02T00:00:00.000Z',
        'Freight': 60.26,
        'ShipName': 'Princesa Isabel Vinhos',
        'ShipAddress': 'Estrada da saúde n. 58',
        'ShipCity': 'Lisboa',
        'ShipRegion': null,
        'ShipCountry': 'Portugal'
    },
    {
        'OrderID': 10398,
        'CustomerID': 'SAVEA',
        'OrderDate': '1996-12-30T00:00:00.000Z',

```

```

    'ShippedDate': '1997-01-09T00:00:00.000Z',
    'Freight': 89.16,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10399,
    'CustomerID': 'VAFFE',
    'OrderDate': '1996-12-31T00:00:00.000Z',
    'ShippedDate': '1997-01-08T00:00:00.000Z',
    'Freight': 27.36,
    'ShipName': 'Vaffeljernet',
    'ShipAddress': 'Smagsloget 45',
    'ShipCity': 'Århus',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10400,
    'CustomerID': 'EASTC',
    'OrderDate': '1997-01-01T00:00:00.000Z',
    'ShippedDate': '1997-01-16T00:00:00.000Z',
    'Freight': 83.93,
    'ShipName': 'Eastern Connection',
    'ShipAddress': '35 King George',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10401,
    'CustomerID': 'RATTC',
    'OrderDate': '1997-01-01T00:00:00.000Z',
    'ShippedDate': '1997-01-10T00:00:00.000Z',
    'Freight': 12.51,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10402,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-01-02T00:00:00.000Z',
    'ShippedDate': '1997-01-10T00:00:00.000Z',
    'Freight': 67.88,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {

```

```

    'OrderID': 10403,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-01-03T00:00:00.000Z',
    'ShippedDate': '1997-01-09T00:00:00.000Z',
    'Freight': 73.79,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10404,
    'CustomerID': 'MAGAA',
    'OrderDate': '1997-01-03T00:00:00.000Z',
    'ShippedDate': '1997-01-08T00:00:00.000Z',
    'Freight': 155.97,
    'ShipName': 'Magazzini Alimentari Riuniti',
    'ShipAddress': 'Via Ludovico il Moro 22',
    'ShipCity': 'Bergamo',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10405,
    'CustomerID': 'LINOD',
    'OrderDate': '1997-01-06T00:00:00.000Z',
    'ShippedDate': '1997-01-22T00:00:00.000Z',
    'Freight': 34.82,
    'ShipName': 'LINO-Delicateses',
    'ShipAddress': 'Ave. 5 de Mayo Porlamar',
    'ShipCity': 'I. de Margarita',
    'ShipRegion': 'Nueva Esparta',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10406,
    'CustomerID': 'QUEEN',
    'OrderDate': '1997-01-07T00:00:00.000Z',
    'ShippedDate': '1997-01-13T00:00:00.000Z',
    'Freight': 108.04,
    'ShipName': 'Queen Cozinha',
    'ShipAddress': 'Alameda dos Canários, 891',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10407,
    'CustomerID': 'OTTIK',
    'OrderDate': '1997-01-07T00:00:00.000Z',
    'ShippedDate': '1997-01-30T00:00:00.000Z',
    'Freight': 91.48,
    'ShipName': 'Ottilies Käseladen',
    'ShipAddress': 'Mehrheimerstr. 369',
    'ShipCity': 'Köln',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10408,
    'CustomerID': 'FOLIG',
    'OrderDate': '1997-01-08T00:00:00.000Z',
    'ShippedDate': '1997-01-14T00:00:00.000Z',
    'Freight': 11.26,
    'ShipName': 'Folies gourmandes',
    'ShipAddress': '184, chaussée de Tournai',
    'ShipCity': 'Lille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10409,
    'CustomerID': 'OCEAN',
    'OrderDate': '1997-01-09T00:00:00.000Z',
    'ShippedDate': '1997-01-14T00:00:00.000Z',
    'Freight': 29.83,
    'ShipName': 'Océano Atlántico Ltda.',
    'ShipAddress': 'Ing. Gustavo Moncada 8585 Piso 20-A',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 10410,
    'CustomerID': 'BOTTM',
    'OrderDate': '1997-01-10T00:00:00.000Z',
    'ShippedDate': '1997-01-15T00:00:00.000Z',
    'Freight': 2.4,
    'ShipName': 'Bottom-Dollar Markets',
    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10411,
    'CustomerID': 'BOTTM',
    'OrderDate': '1997-01-10T00:00:00.000Z',
    'ShippedDate': '1997-01-21T00:00:00.000Z',
    'Freight': 23.65,
    'ShipName': 'Bottom-Dollar Markets',
    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10412,
    'CustomerID': 'WARTH',
    'OrderDate': '1997-01-13T00:00:00.000Z',
    'ShippedDate': '1997-01-15T00:00:00.000Z',
    'Freight': 3.77,
    'ShipName': 'Wartian Herkku',

```



```

        'ShipAddress': 'Torikatu 38',
        'ShipCity': 'Oulu',
        'ShipRegion': null,
        'ShipCountry': 'Finland'
    },
    {
        'OrderID': 10413,
        'CustomerID': 'LAMAI',
        'OrderDate': '1997-01-14T00:00:00.000Z',
        'ShippedDate': '1997-01-16T00:00:00.000Z',
        'Freight': 95.66,
        'ShipName': 'La maison d\'Asie',
        'ShipAddress': '1 rue Alsace-Lorraine',
        'ShipCity': 'Toulouse',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10414,
        'CustomerID': 'FAMIA',
        'OrderDate': '1997-01-14T00:00:00.000Z',
        'ShippedDate': '1997-01-17T00:00:00.000Z',
        'Freight': 21.48,
        'ShipName': 'Familia Arquibaldo',
        'ShipAddress': 'Rua Orós, 92',
        'ShipCity': 'Sao Paulo',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10415,
        'CustomerID': 'HUNGC',
        'OrderDate': '1997-01-15T00:00:00.000Z',
        'ShippedDate': '1997-01-24T00:00:00.000Z',
        'Freight': 0.2,
        'ShipName': 'Hungry Coyote Import Store',
        'ShipAddress': 'City Center Plaza 516 Main St.',
        'ShipCity': 'Elgin',
        'ShipRegion': 'OR',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10416,
        'CustomerID': 'WARTH',
        'OrderDate': '1997-01-16T00:00:00.000Z',
        'ShippedDate': '1997-01-27T00:00:00.000Z',
        'Freight': 22.72,
        'ShipName': 'Wartian Herkku',
        'ShipAddress': 'Torikatu 38',
        'ShipCity': 'Oulu',
        'ShipRegion': null,
        'ShipCountry': 'Finland'
    },
    {
        'OrderID': 10417,
        'CustomerID': 'SIMOB',
        'OrderDate': '1997-01-16T00:00:00.000Z',

```

```

    'ShippedDate': '1997-01-28T00:00:00.000Z',
    'Freight': 70.29,
    'ShipName': 'Simons bistro',
    'ShipAddress': 'Vinbæltet 34',
    'ShipCity': 'Kobenhavn',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10418,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-01-17T00:00:00.000Z',
    'ShippedDate': '1997-01-24T00:00:00.000Z',
    'Freight': 17.55,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10419,
    'CustomerID': 'RICSU',
    'OrderDate': '1997-01-20T00:00:00.000Z',
    'ShippedDate': '1997-01-30T00:00:00.000Z',
    'Freight': 137.35,
    'ShipName': 'Richter Supermarkt',
    'ShipAddress': 'Starenweg 5',
    'ShipCity': 'Genève',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10420,
    'CustomerID': 'WELLI',
    'OrderDate': '1997-01-21T00:00:00.000Z',
    'ShippedDate': '1997-01-27T00:00:00.000Z',
    'Freight': 44.12,
    'ShipName': 'Wellington Importadora',
    'ShipAddress': 'Rua do Mercado, 12',
    'ShipCity': 'Resende',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10421,
    'CustomerID': 'QUEDE',
    'OrderDate': '1997-01-21T00:00:00.000Z',
    'ShippedDate': '1997-01-27T00:00:00.000Z',
    'Freight': 99.23,
    'ShipName': 'Que Delícia',
    'ShipAddress': 'Rua da Panificadora, 12',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {

```

```

    'OrderID': 10422,
    'CustomerID': 'FRANS',
    'OrderDate': '1997-01-22T00:00:00.000Z',
    'ShippedDate': '1997-01-31T00:00:00.000Z',
    'Freight': 3.02,
    'ShipName': 'Franchi S.p.A.',
    'ShipAddress': 'Via Monte Bianco 34',
    'ShipCity': 'Torino',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10423,
    'CustomerID': 'GOURL',
    'OrderDate': '1997-01-23T00:00:00.000Z',
    'ShippedDate': '1997-02-24T00:00:00.000Z',
    'Freight': 24.5,
    'ShipName': 'Gourmet Lanchonetes',
    'ShipAddress': 'Av. Brasil, 442',
    'ShipCity': 'Campinas',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10424,
    'CustomerID': 'MEREP',
    'OrderDate': '1997-01-23T00:00:00.000Z',
    'ShippedDate': '1997-01-27T00:00:00.000Z',
    'Freight': 370.61,
    'ShipName': 'Mère Paillarde',
    'ShipAddress': '43 rue St. Laurent',
    'ShipCity': 'Montréal',
    'ShipRegion': 'Québec',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10425,
    'CustomerID': 'LAMAI',
    'OrderDate': '1997-01-24T00:00:00.000Z',
    'ShippedDate': '1997-02-14T00:00:00.000Z',
    'Freight': 7.93,
    'ShipName': 'La maison d\'Asie',
    'ShipAddress': '1 rue Alsace-Lorraine',
    'ShipCity': 'Toulouse',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10426,
    'CustomerID': 'GALED',
    'OrderDate': '1997-01-27T00:00:00.000Z',
    'ShippedDate': '1997-02-06T00:00:00.000Z',
    'Freight': 18.69,
    'ShipName': 'Galería del gastronómo',
    'ShipAddress': 'Rambla de Catalunya, 23',
    'ShipCity': 'Barcelona',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10427,
    'CustomerID': 'PICCO',
    'OrderDate': '1997-01-27T00:00:00.000Z',
    'ShippedDate': '1997-03-03T00:00:00.000Z',
    'Freight': 31.29,
    'ShipName': 'Piccolo und mehr',
    'ShipAddress': 'Geislweg 14',
    'ShipCity': 'Salzburg',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10428,
    'CustomerID': 'REGGC',
    'OrderDate': '1997-01-28T00:00:00.000Z',
    'ShippedDate': '1997-02-04T00:00:00.000Z',
    'Freight': 11.09,
    'ShipName': 'Reggiani Caseifici',
    'ShipAddress': 'Strada Provinciale 124',
    'ShipCity': 'Reggio Emilia',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10429,
    'CustomerID': 'HUNGO',
    'OrderDate': '1997-01-29T00:00:00.000Z',
    'ShippedDate': '1997-02-07T00:00:00.000Z',
    'Freight': 56.63,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10430,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-01-30T00:00:00.000Z',
    'ShippedDate': '1997-02-03T00:00:00.000Z',
    'Freight': 458.78,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10431,
    'CustomerID': 'BOTTM',
    'OrderDate': '1997-01-30T00:00:00.000Z',
    'ShippedDate': '1997-02-07T00:00:00.000Z',
    'Freight': 44.17,
    'ShipName': 'Bottom-Dollar Markets',

```

```

    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10432,
    'CustomerID': 'SPLIR',
    'OrderDate': '1997-01-31T00:00:00.000Z',
    'ShippedDate': '1997-02-07T00:00:00.000Z',
    'Freight': 4.34,
    'ShipName': 'Split Rail Beer & Ale',
    'ShipAddress': 'P.O. Box 555',
    'ShipCity': 'Lander',
    'ShipRegion': 'WY',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10433,
    'CustomerID': 'PRINI',
    'OrderDate': '1997-02-03T00:00:00.000Z',
    'ShippedDate': '1997-03-04T00:00:00.000Z',
    'Freight': 73.83,
    'ShipName': 'Princesa Isabel Vinhos',
    'ShipAddress': 'Estrada da saúde n. 58',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,
    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 10434,
    'CustomerID': 'FOLKO',
    'OrderDate': '1997-02-03T00:00:00.000Z',
    'ShippedDate': '1997-02-13T00:00:00.000Z',
    'Freight': 17.92,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10435,
    'CustomerID': 'CONSH',
    'OrderDate': '1997-02-04T00:00:00.000Z',
    'ShippedDate': '1997-02-07T00:00:00.000Z',
    'Freight': 9.21,
    'ShipName': 'Consolidated Holdings',
    'ShipAddress': 'Berkeley Gardens 12 Brewery',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10436,
    'CustomerID': 'BLONP',
    'OrderDate': '1997-02-05T00:00:00.000Z',

```

```

        'ShippedDate': '1997-02-11T00:00:00.000Z',
        'Freight': 156.66,
        'ShipName': 'Blondel père et fils',
        'ShipAddress': '24, place Kléber',
        'ShipCity': 'Strasbourg',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10437,
        'CustomerID': 'WARTH',
        'OrderDate': '1997-02-05T00:00:00.000Z',
        'ShippedDate': '1997-02-12T00:00:00.000Z',
        'Freight': 19.97,
        'ShipName': 'Wartian Herkku',
        'ShipAddress': 'Torikatu 38',
        'ShipCity': 'Oulu',
        'ShipRegion': null,
        'ShipCountry': 'Finland'
    },
    {
        'OrderID': 10438,
        'CustomerID': 'TOMSP',
        'OrderDate': '1997-02-06T00:00:00.000Z',
        'ShippedDate': '1997-02-14T00:00:00.000Z',
        'Freight': 8.24,
        'ShipName': 'Toms Spezialitäten',
        'ShipAddress': 'Luisenstr. 48',
        'ShipCity': 'Münster',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10439,
        'CustomerID': 'MEREP',
        'OrderDate': '1997-02-07T00:00:00.000Z',
        'ShippedDate': '1997-02-10T00:00:00.000Z',
        'Freight': 4.07,
        'ShipName': 'Mère Paillarde',
        'ShipAddress': '43 rue St. Laurent',
        'ShipCity': 'Montréal',
        'ShipRegion': 'Québec',
        'ShipCountry': 'Canada'
    },
    {
        'OrderID': 10440,
        'CustomerID': 'SAVEA',
        'OrderDate': '1997-02-10T00:00:00.000Z',
        'ShippedDate': '1997-02-28T00:00:00.000Z',
        'Freight': 86.53,
        'ShipName': 'Save-a-lot Markets',
        'ShipAddress': '187 Suffolk Ln.',
        'ShipCity': 'Boise',
        'ShipRegion': 'ID',
        'ShipCountry': 'USA'
    }
]

```

```

    'OrderID': 10441,
    'CustomerID': 'OLDWO',
    'OrderDate': '1997-02-10T00:00:00.000Z',
    'ShippedDate': '1997-03-14T00:00:00.000Z',
    'Freight': 73.02,
    'ShipName': 'Old World Delicatessen',
    'ShipAddress': '2743 Bering St.',
    'ShipCity': 'Anchorage',
    'ShipRegion': 'AK',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10442,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-02-11T00:00:00.000Z',
    'ShippedDate': '1997-02-18T00:00:00.000Z',
    'Freight': 47.94,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10443,
    'CustomerID': 'REGGC',
    'OrderDate': '1997-02-12T00:00:00.000Z',
    'ShippedDate': '1997-02-14T00:00:00.000Z',
    'Freight': 13.95,
    'ShipName': 'Reggiani Caseifici',
    'ShipAddress': 'Strada Provinciale 124',
    'ShipCity': 'Reggio Emilia',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10444,
    'CustomerID': 'BERGS',
    'OrderDate': '1997-02-12T00:00:00.000Z',
    'ShippedDate': '1997-02-21T00:00:00.000Z',
    'Freight': 3.5,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10445,
    'CustomerID': 'BERGS',
    'OrderDate': '1997-02-13T00:00:00.000Z',
    'ShippedDate': '1997-02-20T00:00:00.000Z',
    'Freight': 9.3,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,

```

```

        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 10446,
        'CustomerID': 'TOMSP',
        'OrderDate': '1997-02-14T00:00:00.000Z',
        'ShippedDate': '1997-02-19T00:00:00.000Z',
        'Freight': 14.68,
        'ShipName': 'Toms Spezialitäten',
        'ShipAddress': 'Luisenstr. 48',
        'ShipCity': 'Münster',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10447,
        'CustomerID': 'RICAR',
        'OrderDate': '1997-02-14T00:00:00.000Z',
        'ShippedDate': '1997-03-07T00:00:00.000Z',
        'Freight': 68.66,
        'ShipName': 'Ricardo Adocicados',
        'ShipAddress': 'Av. Copacabana, 267',
        'ShipCity': 'Rio de Janeiro',
        'ShipRegion': 'RJ',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10448,
        'CustomerID': 'RANCH',
        'OrderDate': '1997-02-17T00:00:00.000Z',
        'ShippedDate': '1997-02-24T00:00:00.000Z',
        'Freight': 38.82,
        'ShipName': 'Rancho grande',
        'ShipAddress': 'Av. del Libertador 900',
        'ShipCity': 'Buenos Aires',
        'ShipRegion': null,
        'ShipCountry': 'Argentina'
    },
    {
        'OrderID': 10449,
        'CustomerID': 'BLONP',
        'OrderDate': '1997-02-18T00:00:00.000Z',
        'ShippedDate': '1997-02-27T00:00:00.000Z',
        'Freight': 53.3,
        'ShipName': 'Blondel père et fils',
        'ShipAddress': '24, place Kléber',
        'ShipCity': 'Strasbourg',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10450,
        'CustomerID': 'VICTE',
        'OrderDate': '1997-02-19T00:00:00.000Z',
        'ShippedDate': '1997-03-11T00:00:00.000Z',
        'Freight': 7.23,
        'ShipName': 'Victuailles en stock',

```



```

    'ShipAddress': '2, rue du Commerce',
    'ShipCity': 'Lyon',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10451,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-02-19T00:00:00.000Z',
    'ShippedDate': '1997-03-12T00:00:00.000Z',
    'Freight': 189.09,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10452,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-02-20T00:00:00.000Z',
    'ShippedDate': '1997-02-26T00:00:00.000Z',
    'Freight': 140.26,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10453,
    'CustomerID': 'AROUT',
    'OrderDate': '1997-02-21T00:00:00.000Z',
    'ShippedDate': '1997-02-26T00:00:00.000Z',
    'Freight': 25.36,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10454,
    'CustomerID': 'LAMAI',
    'OrderDate': '1997-02-21T00:00:00.000Z',
    'ShippedDate': '1997-02-25T00:00:00.000Z',
    'Freight': 2.74,
    'ShipName': 'La maison d\'Asie',
    'ShipAddress': '1 rue Alsace-Lorraine',
    'ShipCity': 'Toulouse',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10455,
    'CustomerID': 'WARTH',
    'OrderDate': '1997-02-24T00:00:00.000Z',

```

```

        'ShippedDate': '1997-03-03T00:00:00.000Z',
        'Freight': 180.45,
        'ShipName': 'Wartian Herkku',
        'ShipAddress': 'Torikatu 38',
        'ShipCity': 'Oulu',
        'ShipRegion': null,
        'ShipCountry': 'Finland'
    },
    {
        'OrderID': 10456,
        'CustomerID': 'KOENE',
        'OrderDate': '1997-02-25T00:00:00.000Z',
        'ShippedDate': '1997-02-28T00:00:00.000Z',
        'Freight': 8.12,
        'ShipName': 'Königlich Essen',
        'ShipAddress': 'Maubelstr. 90',
        'ShipCity': 'Brandenburg',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10457,
        'CustomerID': 'KOENE',
        'OrderDate': '1997-02-25T00:00:00.000Z',
        'ShippedDate': '1997-03-03T00:00:00.000Z',
        'Freight': 11.57,
        'ShipName': 'Königlich Essen',
        'ShipAddress': 'Maubelstr. 90',
        'ShipCity': 'Brandenburg',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10458,
        'CustomerID': 'SUPRD',
        'OrderDate': '1997-02-26T00:00:00.000Z',
        'ShippedDate': '1997-03-04T00:00:00.000Z',
        'Freight': 147.06,
        'ShipName': 'Suprêmes délices',
        'ShipAddress': 'Boulevard Tirou, 255',
        'ShipCity': 'Charleroi',
        'ShipRegion': null,
        'ShipCountry': 'Belgium'
    },
    {
        'OrderID': 10459,
        'CustomerID': 'VICTE',
        'OrderDate': '1997-02-27T00:00:00.000Z',
        'ShippedDate': '1997-02-28T00:00:00.000Z',
        'Freight': 25.09,
        'ShipName': 'Victuailles en stock',
        'ShipAddress': '2, rue du Commerce',
        'ShipCity': 'Lyon',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {

```

```

    'OrderID': 10460,
    'CustomerID': 'FOLKO',
    'OrderDate': '1997-02-28T00:00:00.000Z',
    'ShippedDate': '1997-03-03T00:00:00.000Z',
    'Freight': 16.27,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10461,
    'CustomerID': 'LILAS',
    'OrderDate': '1997-02-28T00:00:00.000Z',
    'ShippedDate': '1997-03-05T00:00:00.000Z',
    'Freight': 148.61,
    'ShipName': 'LILA-Supermercado',
    'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10462,
    'CustomerID': 'CONSH',
    'OrderDate': '1997-03-03T00:00:00.000Z',
    'ShippedDate': '1997-03-18T00:00:00.000Z',
    'Freight': 6.17,
    'ShipName': 'Consolidated Holdings',
    'ShipAddress': 'Berkeley Gardens 12 Brewery',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10463,
    'CustomerID': 'SUPRD',
    'OrderDate': '1997-03-04T00:00:00.000Z',
    'ShippedDate': '1997-03-06T00:00:00.000Z',
    'Freight': 14.78,
    'ShipName': 'Suprêmes délices',
    'ShipAddress': 'Boulevard Tirou, 255',
    'ShipCity': 'Charleroi',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10464,
    'CustomerID': 'FURIB',
    'OrderDate': '1997-03-04T00:00:00.000Z',
    'ShippedDate': '1997-03-14T00:00:00.000Z',
    'Freight': 89,
    'ShipName': 'Furia Bacalhau e Frutos do Mar',
    'ShipAddress': 'Jardim das rosas n. 32',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 10465,
    'CustomerID': 'VAFFE',
    'OrderDate': '1997-03-05T00:00:00.000Z',
    'ShippedDate': '1997-03-14T00:00:00.000Z',
    'Freight': 145.04,
    'ShipName': 'Vaffeljernet',
    'ShipAddress': 'Smagsloget 45',
    'ShipCity': 'Århus',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10466,
    'CustomerID': 'COMMI',
    'OrderDate': '1997-03-06T00:00:00.000Z',
    'ShippedDate': '1997-03-13T00:00:00.000Z',
    'Freight': 11.93,
    'ShipName': 'Comércio Mineiro',
    'ShipAddress': 'Av. dos Lusíadas, 23',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10467,
    'CustomerID': 'MAGAA',
    'OrderDate': '1997-03-06T00:00:00.000Z',
    'ShippedDate': '1997-03-11T00:00:00.000Z',
    'Freight': 4.93,
    'ShipName': 'Magazzini Alimentari Riuniti',
    'ShipAddress': 'Via Ludovico il Moro 22',
    'ShipCity': 'Bergamo',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10468,
    'CustomerID': 'KOENE',
    'OrderDate': '1997-03-07T00:00:00.000Z',
    'ShippedDate': '1997-03-12T00:00:00.000Z',
    'Freight': 44.12,
    'ShipName': 'Königlich Essen',
    'ShipAddress': 'Maubelstr. 90',
    'ShipCity': 'Brandenburg',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10469,
    'CustomerID': 'WHITC',
    'OrderDate': '1997-03-10T00:00:00.000Z',
    'ShippedDate': '1997-03-14T00:00:00.000Z',
    'Freight': 60.18,
    'ShipName': 'White Clover Markets',

```

```

    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10470,
    'CustomerID': 'BONAP',
    'OrderDate': '1997-03-11T00:00:00.000Z',
    'ShippedDate': '1997-03-14T00:00:00.000Z',
    'Freight': 64.56,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10471,
    'CustomerID': 'BSBEV',
    'OrderDate': '1997-03-11T00:00:00.000Z',
    'ShippedDate': '1997-03-18T00:00:00.000Z',
    'Freight': 45.59,
    'ShipName': 'B\ ' Beverages',
    'ShipAddress': 'Fauntleroy Circus',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10472,
    'CustomerID': 'SEVES',
    'OrderDate': '1997-03-12T00:00:00.000Z',
    'ShippedDate': '1997-03-19T00:00:00.000Z',
    'Freight': 4.2,
    'ShipName': 'Seven Seas Imports',
    'ShipAddress': '90 Wadhurst Rd.',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10473,
    'CustomerID': 'ISLAT',
    'OrderDate': '1997-03-13T00:00:00.000Z',
    'ShippedDate': '1997-03-21T00:00:00.000Z',
    'Freight': 16.37,
    'ShipName': 'Island Trading',
    'ShipAddress': 'Garden House Crowther Way',
    'ShipCity': 'Cowes',
    'ShipRegion': 'Isle of Wight',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10474,
    'CustomerID': 'PERIC',
    'OrderDate': '1997-03-13T00:00:00.000Z',

```

```

        'ShippedDate': '1997-03-21T00:00:00.000Z',
        'Freight': 83.49,
        'ShipName': 'Pericles Comidas clásicas',
        'ShipAddress': 'Calle Dr. Jorge Cash 321',
        'ShipCity': 'México D.F.',
        'ShipRegion': null,
        'ShipCountry': 'Mexico'
    },
    {
        'OrderID': 10475,
        'CustomerID': 'SUPRD',
        'OrderDate': '1997-03-14T00:00:00.000Z',
        'ShippedDate': '1997-04-04T00:00:00.000Z',
        'Freight': 68.52,
        'ShipName': 'Suprêmes délices',
        'ShipAddress': 'Boulevard Tirou, 255',
        'ShipCity': 'Charleroi',
        'ShipRegion': null,
        'ShipCountry': 'Belgium'
    },
    {
        'OrderID': 10476,
        'CustomerID': 'HILAA',
        'OrderDate': '1997-03-17T00:00:00.000Z',
        'ShippedDate': '1997-03-24T00:00:00.000Z',
        'Freight': 4.41,
        'ShipName': 'HILARION-Abastos',
        'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
        'ShipCity': 'San Cristóbal',
        'ShipRegion': 'Táchira',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10477,
        'CustomerID': 'PRINI',
        'OrderDate': '1997-03-17T00:00:00.000Z',
        'ShippedDate': '1997-03-25T00:00:00.000Z',
        'Freight': 13.02,
        'ShipName': 'Princesa Isabel Vinhos',
        'ShipAddress': 'Estrada da saúde n. 58',
        'ShipCity': 'Lisboa',
        'ShipRegion': null,
        'ShipCountry': 'Portugal'
    },
    {
        'OrderID': 10478,
        'CustomerID': 'VICTE',
        'OrderDate': '1997-03-18T00:00:00.000Z',
        'ShippedDate': '1997-03-26T00:00:00.000Z',
        'Freight': 4.81,
        'ShipName': 'Victuailles en stock',
        'ShipAddress': '2, rue du Commerce',
        'ShipCity': 'Lyon',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {

```

```

    'OrderID': 10479,
    'CustomerID': 'RATTC',
    'OrderDate': '1997-03-19T00:00:00.000Z',
    'ShippedDate': '1997-03-21T00:00:00.000Z',
    'Freight': 708.95,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10480,
    'CustomerID': 'FOLIG',
    'OrderDate': '1997-03-20T00:00:00.000Z',
    'ShippedDate': '1997-03-24T00:00:00.000Z',
    'Freight': 1.35,
    'ShipName': 'Folies gourmandes',
    'ShipAddress': '184, chaussée de Tournai',
    'ShipCity': 'Lille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10481,
    'CustomerID': 'RICAR',
    'OrderDate': '1997-03-20T00:00:00.000Z',
    'ShippedDate': '1997-03-25T00:00:00.000Z',
    'Freight': 64.33,
    'ShipName': 'Ricardo Adocicados',
    'ShipAddress': 'Av. Copacabana, 267',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10482,
    'CustomerID': 'LAZYK',
    'OrderDate': '1997-03-21T00:00:00.000Z',
    'ShippedDate': '1997-04-10T00:00:00.000Z',
    'Freight': 7.48,
    'ShipName': 'Lazy K Kountry Store',
    'ShipAddress': '12 Orchestra Terrace',
    'ShipCity': 'Walla Walla',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10483,
    'CustomerID': 'WHITC',
    'OrderDate': '1997-03-24T00:00:00.000Z',
    'ShippedDate': '1997-04-25T00:00:00.000Z',
    'Freight': 15.28,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',

```

```

        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10484,
        'CustomerID': 'BSBEV',
        'OrderDate': '1997-03-24T00:00:00.000Z',
        'ShippedDate': '1997-04-01T00:00:00.000Z',
        'Freight': 6.88,
        'ShipName': 'B\ Beverages',
        'ShipAddress': 'Fauntleroy Circus',
        'ShipCity': 'London',
        'ShipRegion': null,
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10485,
        'CustomerID': 'LINOD',
        'OrderDate': '1997-03-25T00:00:00.000Z',
        'ShippedDate': '1997-03-31T00:00:00.000Z',
        'Freight': 64.45,
        'ShipName': 'LINO-Delicateses',
        'ShipAddress': 'Ave. 5 de Mayo Porlamar',
        'ShipCity': 'I. de Margarita',
        'ShipRegion': 'Nueva Esparta',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10486,
        'CustomerID': 'HILAA',
        'OrderDate': '1997-03-26T00:00:00.000Z',
        'ShippedDate': '1997-04-02T00:00:00.000Z',
        'Freight': 30.53,
        'ShipName': 'HILARION-Abastos',
        'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
        'ShipCity': 'San Cristóbal',
        'ShipRegion': 'Táchira',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10487,
        'CustomerID': 'QUEEN',
        'OrderDate': '1997-03-26T00:00:00.000Z',
        'ShippedDate': '1997-03-28T00:00:00.000Z',
        'Freight': 71.07,
        'ShipName': 'Queen Cozinha',
        'ShipAddress': 'Alameda dos Canários, 891',
        'ShipCity': 'Sao Paulo',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10488,
        'CustomerID': 'FRANK',
        'OrderDate': '1997-03-27T00:00:00.000Z',
        'ShippedDate': '1997-04-02T00:00:00.000Z',
        'Freight': 4.93,
        'ShipName': 'Frankenversand',

```



```

    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10489,
    'CustomerID': 'PICCO',
    'OrderDate': '1997-03-28T00:00:00.000Z',
    'ShippedDate': '1997-04-09T00:00:00.000Z',
    'Freight': 5.29,
    'ShipName': 'Piccolo und mehr',
    'ShipAddress': 'Geislweg 14',
    'ShipCity': 'Salzburg',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10490,
    'CustomerID': 'HILAA',
    'OrderDate': '1997-03-31T00:00:00.000Z',
    'ShippedDate': '1997-04-03T00:00:00.000Z',
    'Freight': 210.19,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10491,
    'CustomerID': 'FURIB',
    'OrderDate': '1997-03-31T00:00:00.000Z',
    'ShippedDate': '1997-04-08T00:00:00.000Z',
    'Freight': 16.96,
    'ShipName': 'Furia Bacalhau e Frutos do Mar',
    'ShipAddress': 'Jardim das rosas n. 32',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,
    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 10492,
    'CustomerID': 'BOTTM',
    'OrderDate': '1997-04-01T00:00:00.000Z',
    'ShippedDate': '1997-04-11T00:00:00.000Z',
    'Freight': 62.89,
    'ShipName': 'Bottom-Dollar Markets',
    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10493,
    'CustomerID': 'LAMAI',
    'OrderDate': '1997-04-02T00:00:00.000Z',

```

```

        'ShippedDate': '1997-04-10T00:00:00.000Z',
        'Freight': 10.64,
        'ShipName': 'La maison d\'Asie',
        'ShipAddress': '1 rue Alsace-Lorraine',
        'ShipCity': 'Toulouse',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10494,
        'CustomerID': 'COMMI',
        'OrderDate': '1997-04-02T00:00:00.000Z',
        'ShippedDate': '1997-04-09T00:00:00.000Z',
        'Freight': 65.99,
        'ShipName': 'Comércio Mineiro',
        'ShipAddress': 'Av. dos Lusíadas, 23',
        'ShipCity': 'Sao Paulo',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10495,
        'CustomerID': 'LAUGB',
        'OrderDate': '1997-04-03T00:00:00.000Z',
        'ShippedDate': '1997-04-11T00:00:00.000Z',
        'Freight': 4.65,
        'ShipName': 'Laughing Bacchus Wine Cellars',
        'ShipAddress': '2319 Elm St.',
        'ShipCity': 'Vancouver',
        'ShipRegion': 'BC',
        'ShipCountry': 'Canada'
    },
    {
        'OrderID': 10496,
        'CustomerID': 'TRADH',
        'OrderDate': '1997-04-04T00:00:00.000Z',
        'ShippedDate': '1997-04-07T00:00:00.000Z',
        'Freight': 46.77,
        'ShipName': 'Tradição Hipermercados',
        'ShipAddress': 'Av. Inês de Castro, 414',
        'ShipCity': 'Sao Paulo',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10497,
        'CustomerID': 'LEHMS',
        'OrderDate': '1997-04-04T00:00:00.000Z',
        'ShippedDate': '1997-04-07T00:00:00.000Z',
        'Freight': 36.21,
        'ShipName': 'Lehmanns Marktstand',
        'ShipAddress': 'Magazinweg 7',
        'ShipCity': 'Frankfurt a.M.',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {

```

```

    'OrderID': 10498,
    'CustomerID': 'HILAA',
    'OrderDate': '1997-04-07T00:00:00.000Z',
    'ShippedDate': '1997-04-11T00:00:00.000Z',
    'Freight': 29.75,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10499,
    'CustomerID': 'LILAS',
    'OrderDate': '1997-04-08T00:00:00.000Z',
    'ShippedDate': '1997-04-16T00:00:00.000Z',
    'Freight': 102.02,
    'ShipName': 'LILA-Supermercado',
    'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10500,
    'CustomerID': 'LAMAI',
    'OrderDate': '1997-04-09T00:00:00.000Z',
    'ShippedDate': '1997-04-17T00:00:00.000Z',
    'Freight': 42.68,
    'ShipName': 'La maison d\'Asie',
    'ShipAddress': '1 rue Alsace-Lorraine',
    'ShipCity': 'Toulouse',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10501,
    'CustomerID': 'BLAUS',
    'OrderDate': '1997-04-09T00:00:00.000Z',
    'ShippedDate': '1997-04-16T00:00:00.000Z',
    'Freight': 8.85,
    'ShipName': 'Blauer See Delikatessen',
    'ShipAddress': 'Forsterstr. 57',
    'ShipCity': 'Mannheim',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10502,
    'CustomerID': 'PERIC',
    'OrderDate': '1997-04-10T00:00:00.000Z',
    'ShippedDate': '1997-04-29T00:00:00.000Z',
    'Freight': 69.32,
    'ShipName': 'Pericles Comidas clásicas',
    'ShipAddress': 'Calle Dr. Jorge Cash 321',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10503,
    'CustomerID': 'HUNGO',
    'OrderDate': '1997-04-11T00:00:00.000Z',
    'ShippedDate': '1997-04-16T00:00:00.000Z',
    'Freight': 16.74,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10504,
    'CustomerID': 'WHITC',
    'OrderDate': '1997-04-11T00:00:00.000Z',
    'ShippedDate': '1997-04-18T00:00:00.000Z',
    'Freight': 59.13,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10505,
    'CustomerID': 'MEREP',
    'OrderDate': '1997-04-14T00:00:00.000Z',
    'ShippedDate': '1997-04-21T00:00:00.000Z',
    'Freight': 7.13,
    'ShipName': 'Mère Paillarde',
    'ShipAddress': '43 rue St. Laurent',
    'ShipCity': 'Montréal',
    'ShipRegion': 'Québec',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10506,
    'CustomerID': 'KOENE',
    'OrderDate': '1997-04-15T00:00:00.000Z',
    'ShippedDate': '1997-05-02T00:00:00.000Z',
    'Freight': 21.19,
    'ShipName': 'Königlich Essen',
    'ShipAddress': 'Maubelstr. 90',
    'ShipCity': 'Brandenburg',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10507,
    'CustomerID': 'ANTON',
    'OrderDate': '1997-04-15T00:00:00.000Z',
    'ShippedDate': '1997-04-22T00:00:00.000Z',
    'Freight': 47.45,
    'ShipName': 'Antonio Moreno Taquería',

```

```

        'ShipAddress': 'Mataderos 2312',
        'ShipCity': 'México D.F.',
        'ShipRegion': null,
        'ShipCountry': 'Mexico'
    },
    {
        'OrderID': 10508,
        'CustomerID': 'OTTIK',
        'OrderDate': '1997-04-16T00:00:00.000Z',
        'ShippedDate': '1997-05-13T00:00:00.000Z',
        'Freight': 4.99,
        'ShipName': 'Ottilies Käseladen',
        'ShipAddress': 'Mehrheimerstr. 369',
        'ShipCity': 'Köln',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10509,
        'CustomerID': 'BLAUS',
        'OrderDate': '1997-04-17T00:00:00.000Z',
        'ShippedDate': '1997-04-29T00:00:00.000Z',
        'Freight': 0.15,
        'ShipName': 'Blauer See Delikatessen',
        'ShipAddress': 'Forsterstr. 57',
        'ShipCity': 'Mannheim',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10510,
        'CustomerID': 'SAVEA',
        'OrderDate': '1997-04-18T00:00:00.000Z',
        'ShippedDate': '1997-04-28T00:00:00.000Z',
        'Freight': 367.63,
        'ShipName': 'Save-a-lot Markets',
        'ShipAddress': '187 Suffolk Ln.',
        'ShipCity': 'Boise',
        'ShipRegion': 'ID',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10511,
        'CustomerID': 'BONAP',
        'OrderDate': '1997-04-18T00:00:00.000Z',
        'ShippedDate': '1997-04-21T00:00:00.000Z',
        'Freight': 350.64,
        'ShipName': 'Bon app',
        'ShipAddress': '12, rue des Bouchers',
        'ShipCity': 'Marseille',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10512,
        'CustomerID': 'FAMIA',
        'OrderDate': '1997-04-21T00:00:00.000Z',

```

```

    'ShippedDate': '1997-04-24T00:00:00.000Z',
    'Freight': 3.53,
    'ShipName': 'Familia Arquibaldo',
    'ShipAddress': 'Rua Orós, 92',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10513,
    'CustomerID': 'WANDK',
    'OrderDate': '1997-04-22T00:00:00.000Z',
    'ShippedDate': '1997-04-28T00:00:00.000Z',
    'Freight': 105.65,
    'ShipName': 'Die Wandernde Kuh',
    'ShipAddress': 'Adenauerallee 900',
    'ShipCity': 'Stuttgart',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10514,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-04-22T00:00:00.000Z',
    'ShippedDate': '1997-05-16T00:00:00.000Z',
    'Freight': 789.95,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10515,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-04-23T00:00:00.000Z',
    'ShippedDate': '1997-05-23T00:00:00.000Z',
    'Freight': 204.47,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10516,
    'CustomerID': 'HUNGO',
    'OrderDate': '1997-04-24T00:00:00.000Z',
    'ShippedDate': '1997-05-01T00:00:00.000Z',
    'Freight': 62.78,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {

```

```

    'OrderID': 10517,
    'CustomerID': 'NORTS',
    'OrderDate': '1997-04-24T00:00:00.000Z',
    'ShippedDate': '1997-04-29T00:00:00.000Z',
    'Freight': 32.07,
    'ShipName': 'North/South',
    'ShipAddress': 'South House 300 Queensbridge',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10518,
    'CustomerID': 'TORTU',
    'OrderDate': '1997-04-25T00:00:00.000Z',
    'ShippedDate': '1997-05-05T00:00:00.000Z',
    'Freight': 218.15,
    'ShipName': 'Tortuga Restaurante',
    'ShipAddress': 'Avda. Azteca 123',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10519,
    'CustomerID': 'CHOPS',
    'OrderDate': '1997-04-28T00:00:00.000Z',
    'ShippedDate': '1997-05-01T00:00:00.000Z',
    'Freight': 91.76,
    'ShipName': 'Chop-suey Chinese',
    'ShipAddress': 'Hauptstr. 31',
    'ShipCity': 'Bern',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10520,
    'CustomerID': 'SANTG',
    'OrderDate': '1997-04-29T00:00:00.000Z',
    'ShippedDate': '1997-05-01T00:00:00.000Z',
    'Freight': 13.37,
    'ShipName': 'Santé Gourmet',
    'ShipAddress': 'Erling Skakkes gate 78',
    'ShipCity': 'Stavern',
    'ShipRegion': null,
    'ShipCountry': 'Norway'
  },
  {
    'OrderID': 10521,
    'CustomerID': 'CACTU',
    'OrderDate': '1997-04-29T00:00:00.000Z',
    'ShippedDate': '1997-05-02T00:00:00.000Z',
    'Freight': 17.22,
    'ShipName': 'Cactus Comidas para llevar',
    'ShipAddress': 'Cerrito 333',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 10522,
    'CustomerID': 'LEHMS',
    'OrderDate': '1997-04-30T00:00:00.000Z',
    'ShippedDate': '1997-05-06T00:00:00.000Z',
    'Freight': 45.33,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10523,
    'CustomerID': 'SEVES',
    'OrderDate': '1997-05-01T00:00:00.000Z',
    'ShippedDate': '1997-05-30T00:00:00.000Z',
    'Freight': 77.63,
    'ShipName': 'Seven Seas Imports',
    'ShipAddress': '90 Wadhurst Rd.',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10524,
    'CustomerID': 'BERGS',
    'OrderDate': '1997-05-01T00:00:00.000Z',
    'ShippedDate': '1997-05-07T00:00:00.000Z',
    'Freight': 244.79,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10525,
    'CustomerID': 'BONAP',
    'OrderDate': '1997-05-02T00:00:00.000Z',
    'ShippedDate': '1997-05-23T00:00:00.000Z',
    'Freight': 11.06,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10526,
    'CustomerID': 'WARTH',
    'OrderDate': '1997-05-05T00:00:00.000Z',
    'ShippedDate': '1997-05-15T00:00:00.000Z',
    'Freight': 58.59,
    'ShipName': 'Wartian Herkku',

```



```

    'ShipAddress': 'Torikatu 38',
    'ShipCity': 'Oulu',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10527,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-05-05T00:00:00.000Z',
    'ShippedDate': '1997-05-07T00:00:00.000Z',
    'Freight': 41.9,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10528,
    'CustomerID': 'GREAL',
    'OrderDate': '1997-05-06T00:00:00.000Z',
    'ShippedDate': '1997-05-09T00:00:00.000Z',
    'Freight': 3.35,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10529,
    'CustomerID': 'MAISD',
    'OrderDate': '1997-05-07T00:00:00.000Z',
    'ShippedDate': '1997-05-09T00:00:00.000Z',
    'Freight': 66.69,
    'ShipName': 'Maison Dewey',
    'ShipAddress': 'Rue Joseph-Bens 532',
    'ShipCity': 'Bruxelles',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10530,
    'CustomerID': 'PICCO',
    'OrderDate': '1997-05-08T00:00:00.000Z',
    'ShippedDate': '1997-05-12T00:00:00.000Z',
    'Freight': 339.22,
    'ShipName': 'Piccolo und mehr',
    'ShipAddress': 'Geislweg 14',
    'ShipCity': 'Salzburg',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10531,
    'CustomerID': 'OCEAN',
    'OrderDate': '1997-05-08T00:00:00.000Z',

```

```

    'ShippedDate': '1997-05-19T00:00:00.000Z',
    'Freight': 8.12,
    'ShipName': 'Océano Atlántico Ltda.',
    'ShipAddress': 'Ing. Gustavo Moncada 8585 Piso 20-A',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 10532,
    'CustomerID': 'EASTC',
    'OrderDate': '1997-05-09T00:00:00.000Z',
    'ShippedDate': '1997-05-12T00:00:00.000Z',
    'Freight': 74.46,
    'ShipName': 'Eastern Connection',
    'ShipAddress': '35 King George',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10533,
    'CustomerID': 'FOLKO',
    'OrderDate': '1997-05-12T00:00:00.000Z',
    'ShippedDate': '1997-05-22T00:00:00.000Z',
    'Freight': 188.04,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10534,
    'CustomerID': 'LEHMS',
    'OrderDate': '1997-05-12T00:00:00.000Z',
    'ShippedDate': '1997-05-14T00:00:00.000Z',
    'Freight': 27.94,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10535,
    'CustomerID': 'ANTON',
    'OrderDate': '1997-05-13T00:00:00.000Z',
    'ShippedDate': '1997-05-21T00:00:00.000Z',
    'Freight': 15.64,
    'ShipName': 'Antonio Moreno Taquería',
    'ShipAddress': 'Mataderos 2312',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {

```

```

    'OrderID': 10536,
    'CustomerID': 'LEHMS',
    'OrderDate': '1997-05-14T00:00:00.000Z',
    'ShippedDate': '1997-06-06T00:00:00.000Z',
    'Freight': 58.88,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10537,
    'CustomerID': 'RICSU',
    'OrderDate': '1997-05-14T00:00:00.000Z',
    'ShippedDate': '1997-05-19T00:00:00.000Z',
    'Freight': 78.85,
    'ShipName': 'Richter Supermarkt',
    'ShipAddress': 'Starenweg 5',
    'ShipCity': 'Genève',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10538,
    'CustomerID': 'BSBEV',
    'OrderDate': '1997-05-15T00:00:00.000Z',
    'ShippedDate': '1997-05-16T00:00:00.000Z',
    'Freight': 4.87,
    'ShipName': 'B\ Beverages',
    'ShipAddress': 'Fauntleroy Circus',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10539,
    'CustomerID': 'BSBEV',
    'OrderDate': '1997-05-16T00:00:00.000Z',
    'ShippedDate': '1997-05-23T00:00:00.000Z',
    'Freight': 12.36,
    'ShipName': 'B\ Beverages',
    'ShipAddress': 'Fauntleroy Circus',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10540,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-05-19T00:00:00.000Z',
    'ShippedDate': '1997-06-13T00:00:00.000Z',
    'Freight': 1007.64,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,

```

```

        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10541,
        'CustomerID': 'HANAR',
        'OrderDate': '1997-05-19T00:00:00.000Z',
        'ShippedDate': '1997-05-29T00:00:00.000Z',
        'Freight': 68.65,
        'ShipName': 'Hanari Carnes',
        'ShipAddress': 'Rua do Paço, 67',
        'ShipCity': 'Rio de Janeiro',
        'ShipRegion': 'RJ',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10542,
        'CustomerID': 'KOENE',
        'OrderDate': '1997-05-20T00:00:00.000Z',
        'ShippedDate': '1997-05-26T00:00:00.000Z',
        'Freight': 10.95,
        'ShipName': 'Königlich Essen',
        'ShipAddress': 'Maubelstr. 90',
        'ShipCity': 'Brandenburg',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10543,
        'CustomerID': 'LILAS',
        'OrderDate': '1997-05-21T00:00:00.000Z',
        'ShippedDate': '1997-05-23T00:00:00.000Z',
        'Freight': 48.17,
        'ShipName': 'LILA-Supermercado',
        'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
        'ShipCity': 'Barquisimeto',
        'ShipRegion': 'Lara',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10544,
        'CustomerID': 'LONEP',
        'OrderDate': '1997-05-21T00:00:00.000Z',
        'ShippedDate': '1997-05-30T00:00:00.000Z',
        'Freight': 24.91,
        'ShipName': 'Lonesome Pine Restaurant',
        'ShipAddress': '89 Chiaroscuro Rd.',
        'ShipCity': 'Portland',
        'ShipRegion': 'OR',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10545,
        'CustomerID': 'LAZYK',
        'OrderDate': '1997-05-22T00:00:00.000Z',
        'ShippedDate': '1997-06-26T00:00:00.000Z',
        'Freight': 11.92,
        'ShipName': 'Lazy K Kountry Store',

```

```

    'ShipAddress': '12 Orchestra Terrace',
    'ShipCity': 'Walla Walla',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10546,
    'CustomerID': 'VICTE',
    'OrderDate': '1997-05-23T00:00:00.000Z',
    'ShippedDate': '1997-05-27T00:00:00.000Z',
    'Freight': 194.72,
    'ShipName': 'Victuailles en stock',
    'ShipAddress': '2, rue du Commerce',
    'ShipCity': 'Lyon',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10547,
    'CustomerID': 'SEVES',
    'OrderDate': '1997-05-23T00:00:00.000Z',
    'ShippedDate': '1997-06-02T00:00:00.000Z',
    'Freight': 178.43,
    'ShipName': 'Seven Seas Imports',
    'ShipAddress': '90 Wadhurst Rd.',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10548,
    'CustomerID': 'TOMSP',
    'OrderDate': '1997-05-26T00:00:00.000Z',
    'ShippedDate': '1997-06-02T00:00:00.000Z',
    'Freight': 1.43,
    'ShipName': 'Toms Spezialitäten',
    'ShipAddress': 'Luisenstr. 48',
    'ShipCity': 'Münster',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10549,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-05-27T00:00:00.000Z',
    'ShippedDate': '1997-05-30T00:00:00.000Z',
    'Freight': 171.24,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10550,
    'CustomerID': 'GODOS',
    'OrderDate': '1997-05-28T00:00:00.000Z',

```

```

    'ShippedDate': '1997-06-06T00:00:00.000Z',
    'Freight': 4.32,
    'ShipName': 'Godos Cocina Típica',
    'ShipAddress': 'C/ Romero, 33',
    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10551,
    'CustomerID': 'FURIB',
    'OrderDate': '1997-05-28T00:00:00.000Z',
    'ShippedDate': '1997-06-06T00:00:00.000Z',
    'Freight': 72.95,
    'ShipName': 'Furia Bacalhau e Frutos do Mar',
    'ShipAddress': 'Jardim das rosas n. 32',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,
    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 10552,
    'CustomerID': 'HILAA',
    'OrderDate': '1997-05-29T00:00:00.000Z',
    'ShippedDate': '1997-06-05T00:00:00.000Z',
    'Freight': 83.22,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10553,
    'CustomerID': 'WARTH',
    'OrderDate': '1997-05-30T00:00:00.000Z',
    'ShippedDate': '1997-06-03T00:00:00.000Z',
    'Freight': 149.49,
    'ShipName': 'Wartian Herkku',
    'ShipAddress': 'Torikatu 38',
    'ShipCity': 'Oulu',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10554,
    'CustomerID': 'OTTIK',
    'OrderDate': '1997-05-30T00:00:00.000Z',
    'ShippedDate': '1997-06-05T00:00:00.000Z',
    'Freight': 120.97,
    'ShipName': 'Ottilies Käseladen',
    'ShipAddress': 'Mehrheimerstr. 369',
    'ShipCity': 'Köln',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {

```

```

    'OrderID': 10555,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-06-02T00:00:00.000Z',
    'ShippedDate': '1997-06-04T00:00:00.000Z',
    'Freight': 252.49,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10556,
    'CustomerID': 'SIMOB',
    'OrderDate': '1997-06-03T00:00:00.000Z',
    'ShippedDate': '1997-06-13T00:00:00.000Z',
    'Freight': 9.8,
    'ShipName': 'Simons bistro',
    'ShipAddress': 'Vinbæltet 34',
    'ShipCity': 'Kobenhavn',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10557,
    'CustomerID': 'LEHMS',
    'OrderDate': '1997-06-03T00:00:00.000Z',
    'ShippedDate': '1997-06-06T00:00:00.000Z',
    'Freight': 96.72,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10558,
    'CustomerID': 'AROUT',
    'OrderDate': '1997-06-04T00:00:00.000Z',
    'ShippedDate': '1997-06-10T00:00:00.000Z',
    'Freight': 72.97,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10559,
    'CustomerID': 'BLONP',
    'OrderDate': '1997-06-05T00:00:00.000Z',
    'ShippedDate': '1997-06-13T00:00:00.000Z',
    'Freight': 8.05,
    'ShipName': 'Blondel père et fils',
    'ShipAddress': '24, place Kléber',
    'ShipCity': 'Strasbourg',
    'ShipRegion': null,

```

```

    'ShipCountry': 'France'
  },
  {
    'OrderID': 10560,
    'CustomerID': 'FRANK',
    'OrderDate': '1997-06-06T00:00:00.000Z',
    'ShippedDate': '1997-06-09T00:00:00.000Z',
    'Freight': 36.65,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10561,
    'CustomerID': 'FOLKO',
    'OrderDate': '1997-06-06T00:00:00.000Z',
    'ShippedDate': '1997-06-09T00:00:00.000Z',
    'Freight': 242.21,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10562,
    'CustomerID': 'REGGC',
    'OrderDate': '1997-06-09T00:00:00.000Z',
    'ShippedDate': '1997-06-12T00:00:00.000Z',
    'Freight': 22.95,
    'ShipName': 'Reggiani Caseifici',
    'ShipAddress': 'Strada Provinciale 124',
    'ShipCity': 'Reggio Emilia',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10563,
    'CustomerID': 'RICAR',
    'OrderDate': '1997-06-10T00:00:00.000Z',
    'ShippedDate': '1997-06-24T00:00:00.000Z',
    'Freight': 60.43,
    'ShipName': 'Ricardo Adocicados',
    'ShipAddress': 'Av. Copacabana, 267',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10564,
    'CustomerID': 'RATTC',
    'OrderDate': '1997-06-10T00:00:00.000Z',
    'ShippedDate': '1997-06-16T00:00:00.000Z',
    'Freight': 13.75,
    'ShipName': 'Rattlesnake Canyon Grocery',

```



```

    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10565,
    'CustomerID': 'MEREP',
    'OrderDate': '1997-06-11T00:00:00.000Z',
    'ShippedDate': '1997-06-18T00:00:00.000Z',
    'Freight': 7.15,
    'ShipName': 'Mère Paillarde',
    'ShipAddress': '43 rue St. Laurent',
    'ShipCity': 'Montréal',
    'ShipRegion': 'Québec',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10566,
    'CustomerID': 'BLONP',
    'OrderDate': '1997-06-12T00:00:00.000Z',
    'ShippedDate': '1997-06-18T00:00:00.000Z',
    'Freight': 88.4,
    'ShipName': 'Blondel père et fils',
    'ShipAddress': '24, place Kléber',
    'ShipCity': 'Strasbourg',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10567,
    'CustomerID': 'HUNGO',
    'OrderDate': '1997-06-12T00:00:00.000Z',
    'ShippedDate': '1997-06-17T00:00:00.000Z',
    'Freight': 33.97,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10568,
    'CustomerID': 'GALED',
    'OrderDate': '1997-06-13T00:00:00.000Z',
    'ShippedDate': '1997-07-09T00:00:00.000Z',
    'Freight': 6.54,
    'ShipName': 'Galería del gastronómo',
    'ShipAddress': 'Rambla de Cataluña, 23',
    'ShipCity': 'Barcelona',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10569,
    'CustomerID': 'RATTC',
    'OrderDate': '1997-06-16T00:00:00.000Z',

```

```

    'ShippedDate': '1997-07-11T00:00:00.000Z',
    'Freight': 58.98,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10570,
    'CustomerID': 'MEREP',
    'OrderDate': '1997-06-17T00:00:00.000Z',
    'ShippedDate': '1997-06-19T00:00:00.000Z',
    'Freight': 188.99,
    'ShipName': 'Mère Paillarde',
    'ShipAddress': '43 rue St. Laurent',
    'ShipCity': 'Montréal',
    'ShipRegion': 'Québec',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10571,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-06-17T00:00:00.000Z',
    'ShippedDate': '1997-07-04T00:00:00.000Z',
    'Freight': 26.06,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10572,
    'CustomerID': 'BERGS',
    'OrderDate': '1997-06-18T00:00:00.000Z',
    'ShippedDate': '1997-06-25T00:00:00.000Z',
    'Freight': 116.43,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10573,
    'CustomerID': 'ANTON',
    'OrderDate': '1997-06-19T00:00:00.000Z',
    'ShippedDate': '1997-06-20T00:00:00.000Z',
    'Freight': 84.84,
    'ShipName': 'Antonio Moreno Taquería',
    'ShipAddress': 'Mataderos 2312',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {

```

```

    'OrderID': 10574,
    'CustomerID': 'TRAIH',
    'OrderDate': '1997-06-19T00:00:00.000Z',
    'ShippedDate': '1997-06-30T00:00:00.000Z',
    'Freight': 37.6,
    'ShipName': 'Trail\' Head Gourmet Provisioners',
    'ShipAddress': '722 DaVinci Blvd.',
    'ShipCity': 'Kirkland',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10575,
    'CustomerID': 'MORGK',
    'OrderDate': '1997-06-20T00:00:00.000Z',
    'ShippedDate': '1997-06-30T00:00:00.000Z',
    'Freight': 127.34,
    'ShipName': 'Morgenstern Gesundkost',
    'ShipAddress': 'Heerstr. 22',
    'ShipCity': 'Leipzig',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10576,
    'CustomerID': 'TORTU',
    'OrderDate': '1997-06-23T00:00:00.000Z',
    'ShippedDate': '1997-06-30T00:00:00.000Z',
    'Freight': 18.56,
    'ShipName': 'Tortuga Restaurante',
    'ShipAddress': 'Avda. Azteca 123',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10577,
    'CustomerID': 'TRAIH',
    'OrderDate': '1997-06-23T00:00:00.000Z',
    'ShippedDate': '1997-06-30T00:00:00.000Z',
    'Freight': 25.41,
    'ShipName': 'Trail\' Head Gourmet Provisioners',
    'ShipAddress': '722 DaVinci Blvd.',
    'ShipCity': 'Kirkland',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10578,
    'CustomerID': 'BSBEV',
    'OrderDate': '1997-06-24T00:00:00.000Z',
    'ShippedDate': '1997-07-25T00:00:00.000Z',
    'Freight': 29.6,
    'ShipName': 'B\' Beverages',
    'ShipAddress': 'Fauntleroy Circus',
    'ShipCity': 'London',
    'ShipRegion': null,

```

```

    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10579,
    'CustomerID': 'LETSS',
    'OrderDate': '1997-06-25T00:00:00.000Z',
    'ShippedDate': '1997-07-04T00:00:00.000Z',
    'Freight': 13.73,
    'ShipName': 'Let\ Stop N Shop',
    'ShipAddress': '87 Polk St. Suite 5',
    'ShipCity': 'San Francisco',
    'ShipRegion': 'CA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10580,
    'CustomerID': 'OTTIK',
    'OrderDate': '1997-06-26T00:00:00.000Z',
    'ShippedDate': '1997-07-01T00:00:00.000Z',
    'Freight': 75.89,
    'ShipName': 'Ottilies Käseladen',
    'ShipAddress': 'Mehrheimerstr. 369',
    'ShipCity': 'Köln',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10581,
    'CustomerID': 'FAMIA',
    'OrderDate': '1997-06-26T00:00:00.000Z',
    'ShippedDate': '1997-07-02T00:00:00.000Z',
    'Freight': 3.01,
    'ShipName': 'Familia Arquibaldo',
    'ShipAddress': 'Rua Orós, 92',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10582,
    'CustomerID': 'BLAUS',
    'OrderDate': '1997-06-27T00:00:00.000Z',
    'ShippedDate': '1997-07-14T00:00:00.000Z',
    'Freight': 27.71,
    'ShipName': 'Blauer See Delikatessen',
    'ShipAddress': 'Forsterstr. 57',
    'ShipCity': 'Mannheim',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10583,
    'CustomerID': 'WARTH',
    'OrderDate': '1997-06-30T00:00:00.000Z',
    'ShippedDate': '1997-07-04T00:00:00.000Z',
    'Freight': 7.28,
    'ShipName': 'Wartian Herkku',

```

```

        'ShipAddress': 'Torikatu 38',
        'ShipCity': 'Oulu',
        'ShipRegion': null,
        'ShipCountry': 'Finland'
    },
    {
        'OrderID': 10584,
        'CustomerID': 'BLONP',
        'OrderDate': '1997-06-30T00:00:00.000Z',
        'ShippedDate': '1997-07-04T00:00:00.000Z',
        'Freight': 59.14,
        'ShipName': 'Blondel père et fils',
        'ShipAddress': '24, place Kléber',
        'ShipCity': 'Strasbourg',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10585,
        'CustomerID': 'WELLI',
        'OrderDate': '1997-07-01T00:00:00.000Z',
        'ShippedDate': '1997-07-10T00:00:00.000Z',
        'Freight': 13.41,
        'ShipName': 'Wellington Importadora',
        'ShipAddress': 'Rua do Mercado, 12',
        'ShipCity': 'Resende',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10586,
        'CustomerID': 'REGGC',
        'OrderDate': '1997-07-02T00:00:00.000Z',
        'ShippedDate': '1997-07-09T00:00:00.000Z',
        'Freight': 0.48,
        'ShipName': 'Reggiani Caseifici',
        'ShipAddress': 'Strada Provinciale 124',
        'ShipCity': 'Reggio Emilia',
        'ShipRegion': null,
        'ShipCountry': 'Italy'
    },
    {
        'OrderID': 10587,
        'CustomerID': 'QUEDE',
        'OrderDate': '1997-07-02T00:00:00.000Z',
        'ShippedDate': '1997-07-09T00:00:00.000Z',
        'Freight': 62.52,
        'ShipName': 'Que Delícia',
        'ShipAddress': 'Rua da Panificadora, 12',
        'ShipCity': 'Rio de Janeiro',
        'ShipRegion': 'RJ',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10588,
        'CustomerID': 'QUICK',
        'OrderDate': '1997-07-03T00:00:00.000Z',

```

```

    'ShippedDate': '1997-07-10T00:00:00.000Z',
    'Freight': 194.67,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10589,
    'CustomerID': 'GREAL',
    'OrderDate': '1997-07-04T00:00:00.000Z',
    'ShippedDate': '1997-07-14T00:00:00.000Z',
    'Freight': 4.42,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10590,
    'CustomerID': 'MEREP',
    'OrderDate': '1997-07-07T00:00:00.000Z',
    'ShippedDate': '1997-07-14T00:00:00.000Z',
    'Freight': 44.77,
    'ShipName': 'Mère Paillarde',
    'ShipAddress': '43 rue St. Laurent',
    'ShipCity': 'Montréal',
    'ShipRegion': 'Québec',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10591,
    'CustomerID': 'VAFFE',
    'OrderDate': '1997-07-07T00:00:00.000Z',
    'ShippedDate': '1997-07-16T00:00:00.000Z',
    'Freight': 55.92,
    'ShipName': 'Vaffeljernet',
    'ShipAddress': 'Smagsloget 45',
    'ShipCity': 'Århus',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10592,
    'CustomerID': 'LEHMS',
    'OrderDate': '1997-07-08T00:00:00.000Z',
    'ShippedDate': '1997-07-16T00:00:00.000Z',
    'Freight': 32.1,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {

```

```

    'OrderID': 10593,
    'CustomerID': 'LEHMS',
    'OrderDate': '1997-07-09T00:00:00.000Z',
    'ShippedDate': '1997-08-13T00:00:00.000Z',
    'Freight': 174.2,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10594,
    'CustomerID': 'OLDWO',
    'OrderDate': '1997-07-09T00:00:00.000Z',
    'ShippedDate': '1997-07-16T00:00:00.000Z',
    'Freight': 5.24,
    'ShipName': 'Old World Delicatessen',
    'ShipAddress': '2743 Bering St.',
    'ShipCity': 'Anchorage',
    'ShipRegion': 'AK',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10595,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-07-10T00:00:00.000Z',
    'ShippedDate': '1997-07-14T00:00:00.000Z',
    'Freight': 96.78,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10596,
    'CustomerID': 'WHITC',
    'OrderDate': '1997-07-11T00:00:00.000Z',
    'ShippedDate': '1997-08-12T00:00:00.000Z',
    'Freight': 16.34,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10597,
    'CustomerID': 'PICCO',
    'OrderDate': '1997-07-11T00:00:00.000Z',
    'ShippedDate': '1997-07-18T00:00:00.000Z',
    'Freight': 35.12,
    'ShipName': 'Piccolo und mehr',
    'ShipAddress': 'Geislweg 14',
    'ShipCity': 'Salzburg',
    'ShipRegion': null,

```

```

        'ShipCountry': 'Austria'
    },
    {
        'OrderID': 10598,
        'CustomerID': 'RATTC',
        'OrderDate': '1997-07-14T00:00:00.000Z',
        'ShippedDate': '1997-07-18T00:00:00.000Z',
        'Freight': 44.42,
        'ShipName': 'Rattlesnake Canyon Grocery',
        'ShipAddress': '2817 Milton Dr.',
        'ShipCity': 'Albuquerque',
        'ShipRegion': 'NM',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10599,
        'CustomerID': 'BSBEV',
        'OrderDate': '1997-07-15T00:00:00.000Z',
        'ShippedDate': '1997-07-21T00:00:00.000Z',
        'Freight': 29.98,
        'ShipName': 'B Beverages',
        'ShipAddress': 'Fauntleroy Circus',
        'ShipCity': 'London',
        'ShipRegion': null,
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10600,
        'CustomerID': 'HUNGC',
        'OrderDate': '1997-07-16T00:00:00.000Z',
        'ShippedDate': '1997-07-21T00:00:00.000Z',
        'Freight': 45.13,
        'ShipName': 'Hungry Coyote Import Store',
        'ShipAddress': 'City Center Plaza 516 Main St.',
        'ShipCity': 'Elgin',
        'ShipRegion': 'OR',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10601,
        'CustomerID': 'HILAA',
        'OrderDate': '1997-07-16T00:00:00.000Z',
        'ShippedDate': '1997-07-22T00:00:00.000Z',
        'Freight': 58.3,
        'ShipName': 'HILARION-Abastos',
        'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
        'ShipCity': 'San Cristóbal',
        'ShipRegion': 'Táchira',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10602,
        'CustomerID': 'VAFFE',
        'OrderDate': '1997-07-17T00:00:00.000Z',
        'ShippedDate': '1997-07-22T00:00:00.000Z',
        'Freight': 2.92,
        'ShipName': 'Vaffeljernet',
    }

```



```

        'ShipAddress': 'Smagsloget 45',
        'ShipCity': 'Århus',
        'ShipRegion': null,
        'ShipCountry': 'Denmark'
    },
    {
        'OrderID': 10603,
        'CustomerID': 'SAVEA',
        'OrderDate': '1997-07-18T00:00:00.000Z',
        'ShippedDate': '1997-08-08T00:00:00.000Z',
        'Freight': 48.77,
        'ShipName': 'Save-a-lot Markets',
        'ShipAddress': '187 Suffolk Ln.',
        'ShipCity': 'Boise',
        'ShipRegion': 'ID',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10604,
        'CustomerID': 'FURIB',
        'OrderDate': '1997-07-18T00:00:00.000Z',
        'ShippedDate': '1997-07-29T00:00:00.000Z',
        'Freight': 7.46,
        'ShipName': 'Furia Bacalhau e Frutos do Mar',
        'ShipAddress': 'Jardim das rosas n. 32',
        'ShipCity': 'Lisboa',
        'ShipRegion': null,
        'ShipCountry': 'Portugal'
    },
    {
        'OrderID': 10605,
        'CustomerID': 'MEREP',
        'OrderDate': '1997-07-21T00:00:00.000Z',
        'ShippedDate': '1997-07-29T00:00:00.000Z',
        'Freight': 379.13,
        'ShipName': 'Mère Paillarde',
        'ShipAddress': '43 rue St. Laurent',
        'ShipCity': 'Montréal',
        'ShipRegion': 'Québec',
        'ShipCountry': 'Canada'
    },
    {
        'OrderID': 10606,
        'CustomerID': 'TRADH',
        'OrderDate': '1997-07-22T00:00:00.000Z',
        'ShippedDate': '1997-07-31T00:00:00.000Z',
        'Freight': 79.4,
        'ShipName': 'Tradição Hipermercados',
        'ShipAddress': 'Av. Inês de Castro, 414',
        'ShipCity': 'Sao Paulo',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10607,
        'CustomerID': 'SAVEA',
        'OrderDate': '1997-07-22T00:00:00.000Z',

```

```

    'ShippedDate': '1997-07-25T00:00:00.000Z',
    'Freight': 200.24,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10608,
    'CustomerID': 'TOMSP',
    'OrderDate': '1997-07-23T00:00:00.000Z',
    'ShippedDate': '1997-08-01T00:00:00.000Z',
    'Freight': 27.79,
    'ShipName': 'Toms Spezialitäten',
    'ShipAddress': 'Luisenstr. 48',
    'ShipCity': 'Münster',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10609,
    'CustomerID': 'DUMON',
    'OrderDate': '1997-07-24T00:00:00.000Z',
    'ShippedDate': '1997-07-30T00:00:00.000Z',
    'Freight': 1.85,
    'ShipName': 'Du monde entier',
    'ShipAddress': '67, rue des Cinquante Otages',
    'ShipCity': 'Nantes',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10610,
    'CustomerID': 'LAMAI',
    'OrderDate': '1997-07-25T00:00:00.000Z',
    'ShippedDate': '1997-08-06T00:00:00.000Z',
    'Freight': 26.78,
    'ShipName': 'La maison d\'Asie',
    'ShipAddress': '1 rue Alsace-Lorraine',
    'ShipCity': 'Toulouse',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10611,
    'CustomerID': 'WOLZA',
    'OrderDate': '1997-07-25T00:00:00.000Z',
    'ShippedDate': '1997-08-01T00:00:00.000Z',
    'Freight': 80.65,
    'ShipName': 'Wolski Zajazd',
    'ShipAddress': 'ul. Filtrowa 68',
    'ShipCity': 'Warszawa',
    'ShipRegion': null,
    'ShipCountry': 'Poland'
  },
  {

```

```

    'OrderID': 10612,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-07-28T00:00:00.000Z',
    'ShippedDate': '1997-08-01T00:00:00.000Z',
    'Freight': 544.08,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10613,
    'CustomerID': 'HILAA',
    'OrderDate': '1997-07-29T00:00:00.000Z',
    'ShippedDate': '1997-08-01T00:00:00.000Z',
    'Freight': 8.11,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10614,
    'CustomerID': 'BLAUS',
    'OrderDate': '1997-07-29T00:00:00.000Z',
    'ShippedDate': '1997-08-01T00:00:00.000Z',
    'Freight': 1.93,
    'ShipName': 'Blauer See Delikatessen',
    'ShipAddress': 'Forsterstr. 57',
    'ShipCity': 'Mannheim',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10615,
    'CustomerID': 'WILMK',
    'OrderDate': '1997-07-30T00:00:00.000Z',
    'ShippedDate': '1997-08-06T00:00:00.000Z',
    'Freight': 0.75,
    'ShipName': 'Wilman Kala',
    'ShipAddress': 'Keskuskatu 45',
    'ShipCity': 'Helsinki',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10616,
    'CustomerID': 'GREAL',
    'OrderDate': '1997-07-31T00:00:00.000Z',
    'ShippedDate': '1997-08-05T00:00:00.000Z',
    'Freight': 116.53,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',

```

```

    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10617,
    'CustomerID': 'GREAL',
    'OrderDate': '1997-07-31T00:00:00.000Z',
    'ShippedDate': '1997-08-04T00:00:00.000Z',
    'Freight': 18.53,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10618,
    'CustomerID': 'MEREP',
    'OrderDate': '1997-08-01T00:00:00.000Z',
    'ShippedDate': '1997-08-08T00:00:00.000Z',
    'Freight': 154.68,
    'ShipName': 'Mère Paillarde',
    'ShipAddress': '43 rue St. Laurent',
    'ShipCity': 'Montréal',
    'ShipRegion': 'Québec',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10619,
    'CustomerID': 'MEREP',
    'OrderDate': '1997-08-04T00:00:00.000Z',
    'ShippedDate': '1997-08-07T00:00:00.000Z',
    'Freight': 91.05,
    'ShipName': 'Mère Paillarde',
    'ShipAddress': '43 rue St. Laurent',
    'ShipCity': 'Montréal',
    'ShipRegion': 'Québec',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10620,
    'CustomerID': 'LAUGB',
    'OrderDate': '1997-08-05T00:00:00.000Z',
    'ShippedDate': '1997-08-14T00:00:00.000Z',
    'Freight': 0.94,
    'ShipName': 'Laughing Bacchus Wine Cellars',
    'ShipAddress': '2319 Elm St.',
    'ShipCity': 'Vancouver',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10621,
    'CustomerID': 'ISLAT',
    'OrderDate': '1997-08-05T00:00:00.000Z',
    'ShippedDate': '1997-08-11T00:00:00.000Z',
    'Freight': 23.73,
    'ShipName': 'Island Trading',

```

```

        'ShipAddress': 'Garden House Crowther Way',
        'ShipCity': 'Cowes',
        'ShipRegion': 'Isle of Wight',
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10622,
        'CustomerID': 'RICAR',
        'OrderDate': '1997-08-06T00:00:00.000Z',
        'ShippedDate': '1997-08-11T00:00:00.000Z',
        'Freight': 50.97,
        'ShipName': 'Ricardo Adocicados',
        'ShipAddress': 'Av. Copacabana, 267',
        'ShipCity': 'Rio de Janeiro',
        'ShipRegion': 'RJ',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10623,
        'CustomerID': 'FRANK',
        'OrderDate': '1997-08-07T00:00:00.000Z',
        'ShippedDate': '1997-08-12T00:00:00.000Z',
        'Freight': 97.18,
        'ShipName': 'Frankenversand',
        'ShipAddress': 'Berliner Platz 43',
        'ShipCity': 'München',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10624,
        'CustomerID': 'THECR',
        'OrderDate': '1997-08-07T00:00:00.000Z',
        'ShippedDate': '1997-08-19T00:00:00.000Z',
        'Freight': 94.8,
        'ShipName': 'The Cracker Box',
        'ShipAddress': '55 Grizzly Peak Rd.',
        'ShipCity': 'Butte',
        'ShipRegion': 'MT',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10625,
        'CustomerID': 'ANATR',
        'OrderDate': '1997-08-08T00:00:00.000Z',
        'ShippedDate': '1997-08-14T00:00:00.000Z',
        'Freight': 43.9,
        'ShipName': 'Ana Trujillo Emparedados y helados',
        'ShipAddress': 'Avda. de la Constitución 2222',
        'ShipCity': 'México D.F.',
        'ShipRegion': null,
        'ShipCountry': 'Mexico'
    },
    {
        'OrderID': 10626,
        'CustomerID': 'BERGS',
        'OrderDate': '1997-08-11T00:00:00.000Z',

```

```

    'ShippedDate': '1997-08-20T00:00:00.000Z',
    'Freight': 138.69,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10627,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-08-11T00:00:00.000Z',
    'ShippedDate': '1997-08-21T00:00:00.000Z',
    'Freight': 107.46,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10628,
    'CustomerID': 'BLONP',
    'OrderDate': '1997-08-12T00:00:00.000Z',
    'ShippedDate': '1997-08-20T00:00:00.000Z',
    'Freight': 30.36,
    'ShipName': 'Blondel père et fils',
    'ShipAddress': '24, place Kléber',
    'ShipCity': 'Strasbourg',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10629,
    'CustomerID': 'GODOS',
    'OrderDate': '1997-08-12T00:00:00.000Z',
    'ShippedDate': '1997-08-20T00:00:00.000Z',
    'Freight': 85.46,
    'ShipName': 'Godos Cocina Típica',
    'ShipAddress': 'C/ Romero, 33',
    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10630,
    'CustomerID': 'KOENE',
    'OrderDate': '1997-08-13T00:00:00.000Z',
    'ShippedDate': '1997-08-19T00:00:00.000Z',
    'Freight': 32.35,
    'ShipName': 'Königlich Essen',
    'ShipAddress': 'Maubelstr. 90',
    'ShipCity': 'Brandenburg',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {

```

```

    'OrderID': 10631,
    'CustomerID': 'LAMAI',
    'OrderDate': '1997-08-14T00:00:00.000Z',
    'ShippedDate': '1997-08-15T00:00:00.000Z',
    'Freight': 0.87,
    'ShipName': 'La maison d\'Asie',
    'ShipAddress': '1 rue Alsace-Lorraine',
    'ShipCity': 'Toulouse',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10632,
    'CustomerID': 'WANDK',
    'OrderDate': '1997-08-14T00:00:00.000Z',
    'ShippedDate': '1997-08-19T00:00:00.000Z',
    'Freight': 41.38,
    'ShipName': 'Die Wandernde Kuh',
    'ShipAddress': 'Adenauerallee 900',
    'ShipCity': 'Stuttgart',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10633,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-08-15T00:00:00.000Z',
    'ShippedDate': '1997-08-18T00:00:00.000Z',
    'Freight': 477.9,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10634,
    'CustomerID': 'FOLIG',
    'OrderDate': '1997-08-15T00:00:00.000Z',
    'ShippedDate': '1997-08-21T00:00:00.000Z',
    'Freight': 487.38,
    'ShipName': 'Folies gourmandes',
    'ShipAddress': '184, chaussée de Tournai',
    'ShipCity': 'Lille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10635,
    'CustomerID': 'MAGAA',
    'OrderDate': '1997-08-18T00:00:00.000Z',
    'ShippedDate': '1997-08-21T00:00:00.000Z',
    'Freight': 47.46,
    'ShipName': 'Magazzini Alimentari Riuniti',
    'ShipAddress': 'Via Ludovico il Moro 22',
    'ShipCity': 'Bergamo',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10636,
    'CustomerID': 'WARTH',
    'OrderDate': '1997-08-19T00:00:00.000Z',
    'ShippedDate': '1997-08-26T00:00:00.000Z',
    'Freight': 1.15,
    'ShipName': 'Wartian Herkku',
    'ShipAddress': 'Torikatu 38',
    'ShipCity': 'Oulu',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10637,
    'CustomerID': 'QUEEN',
    'OrderDate': '1997-08-19T00:00:00.000Z',
    'ShippedDate': '1997-08-26T00:00:00.000Z',
    'Freight': 201.29,
    'ShipName': 'Queen Cozinha',
    'ShipAddress': 'Alameda dos Canários, 891',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10638,
    'CustomerID': 'LINOD',
    'OrderDate': '1997-08-20T00:00:00.000Z',
    'ShippedDate': '1997-09-01T00:00:00.000Z',
    'Freight': 158.44,
    'ShipName': 'LINO-Delicateses',
    'ShipAddress': 'Ave. 5 de Mayo Porlamar',
    'ShipCity': 'I. de Margarita',
    'ShipRegion': 'Nueva Esparta',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10639,
    'CustomerID': 'SANTG',
    'OrderDate': '1997-08-20T00:00:00.000Z',
    'ShippedDate': '1997-08-27T00:00:00.000Z',
    'Freight': 38.64,
    'ShipName': 'Santé Gourmet',
    'ShipAddress': 'Erling Skakkes gate 78',
    'ShipCity': 'Stavern',
    'ShipRegion': null,
    'ShipCountry': 'Norway'
  },
  {
    'OrderID': 10640,
    'CustomerID': 'WANDK',
    'OrderDate': '1997-08-21T00:00:00.000Z',
    'ShippedDate': '1997-08-28T00:00:00.000Z',
    'Freight': 23.55,
    'ShipName': 'Die Wandernde Kuh',

```



```

        'ShipAddress': 'Adenauerallee 900',
        'ShipCity': 'Stuttgart',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10641,
        'CustomerID': 'HILAA',
        'OrderDate': '1997-08-22T00:00:00.000Z',
        'ShippedDate': '1997-08-26T00:00:00.000Z',
        'Freight': 179.61,
        'ShipName': 'HILARION-Abastos',
        'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
        'ShipCity': 'San Cristóbal',
        'ShipRegion': 'Táchira',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10642,
        'CustomerID': 'SIMOB',
        'OrderDate': '1997-08-22T00:00:00.000Z',
        'ShippedDate': '1997-09-05T00:00:00.000Z',
        'Freight': 41.89,
        'ShipName': 'Simons bistro',
        'ShipAddress': 'Vinbæltet 34',
        'ShipCity': 'Kobenhavn',
        'ShipRegion': null,
        'ShipCountry': 'Denmark'
    },
    {
        'OrderID': 10643,
        'CustomerID': 'ALFKI',
        'OrderDate': '1997-08-25T00:00:00.000Z',
        'ShippedDate': '1997-09-02T00:00:00.000Z',
        'Freight': 29.46,
        'ShipName': 'Alfreds Futterkiste',
        'ShipAddress': 'Obere Str. 57',
        'ShipCity': 'Berlin',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10644,
        'CustomerID': 'WELLI',
        'OrderDate': '1997-08-25T00:00:00.000Z',
        'ShippedDate': '1997-09-01T00:00:00.000Z',
        'Freight': 0.14,
        'ShipName': 'Wellington Importadora',
        'ShipAddress': 'Rua do Mercado, 12',
        'ShipCity': 'Resende',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10645,
        'CustomerID': 'HANAR',
        'OrderDate': '1997-08-26T00:00:00.000Z',

```

```

    'ShippedDate': '1997-09-02T00:00:00.000Z',
    'Freight': 12.41,
    'ShipName': 'Hanari Carnes',
    'ShipAddress': 'Rua do Paço, 67',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10646,
    'CustomerID': 'HUNGO',
    'OrderDate': '1997-08-27T00:00:00.000Z',
    'ShippedDate': '1997-09-03T00:00:00.000Z',
    'Freight': 142.33,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10647,
    'CustomerID': 'QUEDE',
    'OrderDate': '1997-08-27T00:00:00.000Z',
    'ShippedDate': '1997-09-03T00:00:00.000Z',
    'Freight': 45.54,
    'ShipName': 'Que Delícia',
    'ShipAddress': 'Rua da Panificadora, 12',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10648,
    'CustomerID': 'RICAR',
    'OrderDate': '1997-08-28T00:00:00.000Z',
    'ShippedDate': '1997-09-09T00:00:00.000Z',
    'Freight': 14.25,
    'ShipName': 'Ricardo Adocicados',
    'ShipAddress': 'Av. Copacabana, 267',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10649,
    'CustomerID': 'MAISD',
    'OrderDate': '1997-08-28T00:00:00.000Z',
    'ShippedDate': '1997-08-29T00:00:00.000Z',
    'Freight': 6.2,
    'ShipName': 'Maison Dewey',
    'ShipAddress': 'Rue Joseph-Bens 532',
    'ShipCity': 'Bruxelles',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {

```

```

    'OrderID': 10650,
    'CustomerID': 'FAMIA',
    'OrderDate': '1997-08-29T00:00:00.000Z',
    'ShippedDate': '1997-09-03T00:00:00.000Z',
    'Freight': 176.81,
    'ShipName': 'Familia Arquibaldo',
    'ShipAddress': 'Rua Orós, 92',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10651,
    'CustomerID': 'WANDK',
    'OrderDate': '1997-09-01T00:00:00.000Z',
    'ShippedDate': '1997-09-11T00:00:00.000Z',
    'Freight': 20.6,
    'ShipName': 'Die Wandernde Kuh',
    'ShipAddress': 'Adenauerallee 900',
    'ShipCity': 'Stuttgart',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10652,
    'CustomerID': 'GOURL',
    'OrderDate': '1997-09-01T00:00:00.000Z',
    'ShippedDate': '1997-09-08T00:00:00.000Z',
    'Freight': 7.14,
    'ShipName': 'Gourmet Lanchonetes',
    'ShipAddress': 'Av. Brasil, 442',
    'ShipCity': 'Campinas',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10653,
    'CustomerID': 'FRANK',
    'OrderDate': '1997-09-02T00:00:00.000Z',
    'ShippedDate': '1997-09-19T00:00:00.000Z',
    'Freight': 93.25,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10654,
    'CustomerID': 'BERGS',
    'OrderDate': '1997-09-02T00:00:00.000Z',
    'ShippedDate': '1997-09-11T00:00:00.000Z',
    'Freight': 55.26,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10655,
    'CustomerID': 'REGGC',
    'OrderDate': '1997-09-03T00:00:00.000Z',
    'ShippedDate': '1997-09-11T00:00:00.000Z',
    'Freight': 4.41,
    'ShipName': 'Reggiani Caseifici',
    'ShipAddress': 'Strada Provinciale 124',
    'ShipCity': 'Reggio Emilia',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10656,
    'CustomerID': 'GREAL',
    'OrderDate': '1997-09-04T00:00:00.000Z',
    'ShippedDate': '1997-09-10T00:00:00.000Z',
    'Freight': 57.15,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10657,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-09-04T00:00:00.000Z',
    'ShippedDate': '1997-09-15T00:00:00.000Z',
    'Freight': 352.69,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10658,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-09-05T00:00:00.000Z',
    'ShippedDate': '1997-09-08T00:00:00.000Z',
    'Freight': 364.15,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10659,
    'CustomerID': 'QUEEN',
    'OrderDate': '1997-09-05T00:00:00.000Z',
    'ShippedDate': '1997-09-10T00:00:00.000Z',
    'Freight': 105.81,
    'ShipName': 'Queen Cozinha',

```

```

        'ShipAddress': 'Alameda dos Canários, 891',
        'ShipCity': 'Sao Paulo',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10660,
        'CustomerID': 'HUNGC',
        'OrderDate': '1997-09-08T00:00:00.000Z',
        'ShippedDate': '1997-10-15T00:00:00.000Z',
        'Freight': 111.29,
        'ShipName': 'Hungry Coyote Import Store',
        'ShipAddress': 'City Center Plaza 516 Main St.',
        'ShipCity': 'Elgin',
        'ShipRegion': 'OR',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10661,
        'CustomerID': 'HUNGO',
        'OrderDate': '1997-09-09T00:00:00.000Z',
        'ShippedDate': '1997-09-15T00:00:00.000Z',
        'Freight': 17.55,
        'ShipName': 'Hungry Owl All-Night Grocers',
        'ShipAddress': '8 Johnstown Road',
        'ShipCity': 'Cork',
        'ShipRegion': 'Co. Cork',
        'ShipCountry': 'Ireland'
    },
    {
        'OrderID': 10662,
        'CustomerID': 'LONEP',
        'OrderDate': '1997-09-09T00:00:00.000Z',
        'ShippedDate': '1997-09-18T00:00:00.000Z',
        'Freight': 1.28,
        'ShipName': 'Lonesome Pine Restaurant',
        'ShipAddress': '89 Chiaroscuro Rd.',
        'ShipCity': 'Portland',
        'ShipRegion': 'OR',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10663,
        'CustomerID': 'BONAP',
        'OrderDate': '1997-09-10T00:00:00.000Z',
        'ShippedDate': '1997-10-03T00:00:00.000Z',
        'Freight': 113.15,
        'ShipName': 'Bon app',
        'ShipAddress': '12, rue des Bouchers',
        'ShipCity': 'Marseille',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10664,
        'CustomerID': 'FURIB',
        'OrderDate': '1997-09-10T00:00:00.000Z',

```

```

    'ShippedDate': '1997-09-19T00:00:00.000Z',
    'Freight': 1.27,
    'ShipName': 'Furia Bacalhau e Frutos do Mar',
    'ShipAddress': 'Jardim das rosas n. 32',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,
    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 10665,
    'CustomerID': 'LONEP',
    'OrderDate': '1997-09-11T00:00:00.000Z',
    'ShippedDate': '1997-09-17T00:00:00.000Z',
    'Freight': 26.31,
    'ShipName': 'Lonesome Pine Restaurant',
    'ShipAddress': '89 Chiaroscuro Rd.',
    'ShipCity': 'Portland',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10666,
    'CustomerID': 'RICSU',
    'OrderDate': '1997-09-12T00:00:00.000Z',
    'ShippedDate': '1997-09-22T00:00:00.000Z',
    'Freight': 232.42,
    'ShipName': 'Richter Supermarkt',
    'ShipAddress': 'Starenweg 5',
    'ShipCity': 'Genève',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10667,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-09-12T00:00:00.000Z',
    'ShippedDate': '1997-09-19T00:00:00.000Z',
    'Freight': 78.09,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10668,
    'CustomerID': 'WANDK',
    'OrderDate': '1997-09-15T00:00:00.000Z',
    'ShippedDate': '1997-09-23T00:00:00.000Z',
    'Freight': 47.22,
    'ShipName': 'Die Wandernde Kuh',
    'ShipAddress': 'Adenauerallee 900',
    'ShipCity': 'Stuttgart',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {

```

```

    'OrderID': 10669,
    'CustomerID': 'SIMOB',
    'OrderDate': '1997-09-15T00:00:00.000Z',
    'ShippedDate': '1997-09-22T00:00:00.000Z',
    'Freight': 24.39,
    'ShipName': 'Simons bistro',
    'ShipAddress': 'Vinbæltet 34',
    'ShipCity': 'Kobenhavn',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10670,
    'CustomerID': 'FRANK',
    'OrderDate': '1997-09-16T00:00:00.000Z',
    'ShippedDate': '1997-09-18T00:00:00.000Z',
    'Freight': 203.48,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10671,
    'CustomerID': 'FRANR',
    'OrderDate': '1997-09-17T00:00:00.000Z',
    'ShippedDate': '1997-09-24T00:00:00.000Z',
    'Freight': 30.34,
    'ShipName': 'France restauration',
    'ShipAddress': '54, rue Royale',
    'ShipCity': 'Nantes',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10672,
    'CustomerID': 'BERGS',
    'OrderDate': '1997-09-17T00:00:00.000Z',
    'ShippedDate': '1997-09-26T00:00:00.000Z',
    'Freight': 95.75,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10673,
    'CustomerID': 'WILMK',
    'OrderDate': '1997-09-18T00:00:00.000Z',
    'ShippedDate': '1997-09-19T00:00:00.000Z',
    'Freight': 22.76,
    'ShipName': 'Wilman Kala',
    'ShipAddress': 'Keskuskatu 45',
    'ShipCity': 'Helsinki',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10674,
    'CustomerID': 'ISLAT',
    'OrderDate': '1997-09-18T00:00:00.000Z',
    'ShippedDate': '1997-09-30T00:00:00.000Z',
    'Freight': 0.9,
    'ShipName': 'Island Trading',
    'ShipAddress': 'Garden House Crowther Way',
    'ShipCity': 'Cowes',
    'ShipRegion': 'Isle of Wight',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10675,
    'CustomerID': 'FRANK',
    'OrderDate': '1997-09-19T00:00:00.000Z',
    'ShippedDate': '1997-09-23T00:00:00.000Z',
    'Freight': 31.85,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10676,
    'CustomerID': 'TORTU',
    'OrderDate': '1997-09-22T00:00:00.000Z',
    'ShippedDate': '1997-09-29T00:00:00.000Z',
    'Freight': 2.01,
    'ShipName': 'Tortuga Restaurante',
    'ShipAddress': 'Avda. Azteca 123',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10677,
    'CustomerID': 'ANTON',
    'OrderDate': '1997-09-22T00:00:00.000Z',
    'ShippedDate': '1997-09-26T00:00:00.000Z',
    'Freight': 4.03,
    'ShipName': 'Antonio Moreno Taquería',
    'ShipAddress': 'Mataderos 2312',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10678,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-09-23T00:00:00.000Z',
    'ShippedDate': '1997-10-16T00:00:00.000Z',
    'Freight': 388.98,
    'ShipName': 'Save-a-lot Markets',

```



```

    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10679,
    'CustomerID': 'BLONP',
    'OrderDate': '1997-09-23T00:00:00.000Z',
    'ShippedDate': '1997-09-30T00:00:00.000Z',
    'Freight': 27.94,
    'ShipName': 'Blondel père et fils',
    'ShipAddress': '24, place Kléber',
    'ShipCity': 'Strasbourg',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10680,
    'CustomerID': 'OLDWO',
    'OrderDate': '1997-09-24T00:00:00.000Z',
    'ShippedDate': '1997-09-26T00:00:00.000Z',
    'Freight': 26.61,
    'ShipName': 'Old World Delicatessen',
    'ShipAddress': '2743 Bering St.',
    'ShipCity': 'Anchorage',
    'ShipRegion': 'AK',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10681,
    'CustomerID': 'GREAL',
    'OrderDate': '1997-09-25T00:00:00.000Z',
    'ShippedDate': '1997-09-30T00:00:00.000Z',
    'Freight': 76.13,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10682,
    'CustomerID': 'ANTON',
    'OrderDate': '1997-09-25T00:00:00.000Z',
    'ShippedDate': '1997-10-01T00:00:00.000Z',
    'Freight': 36.13,
    'ShipName': 'Antonio Moreno Taquería',
    'ShipAddress': 'Mataderos 2312',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10683,
    'CustomerID': 'DUMON',
    'OrderDate': '1997-09-26T00:00:00.000Z',

```

```

    'ShippedDate': '1997-10-01T00:00:00.000Z',
    'Freight': 4.4,
    'ShipName': 'Du monde entier',
    'ShipAddress': '67, rue des Cinquante Otages',
    'ShipCity': 'Nantes',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10684,
    'CustomerID': 'OTTIK',
    'OrderDate': '1997-09-26T00:00:00.000Z',
    'ShippedDate': '1997-09-30T00:00:00.000Z',
    'Freight': 145.63,
    'ShipName': 'Ottilies Käseladen',
    'ShipAddress': 'Mehrheimerstr. 369',
    'ShipCity': 'Köln',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10685,
    'CustomerID': 'GOURL',
    'OrderDate': '1997-09-29T00:00:00.000Z',
    'ShippedDate': '1997-10-03T00:00:00.000Z',
    'Freight': 33.75,
    'ShipName': 'Gourmet Lanchonetes',
    'ShipAddress': 'Av. Brasil, 442',
    'ShipCity': 'Campinas',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10686,
    'CustomerID': 'PICCO',
    'OrderDate': '1997-09-30T00:00:00.000Z',
    'ShippedDate': '1997-10-08T00:00:00.000Z',
    'Freight': 96.5,
    'ShipName': 'Piccolo und mehr',
    'ShipAddress': 'Geislweg 14',
    'ShipCity': 'Salzburg',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10687,
    'CustomerID': 'HUNGO',
    'OrderDate': '1997-09-30T00:00:00.000Z',
    'ShippedDate': '1997-10-30T00:00:00.000Z',
    'Freight': 296.43,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {

```

```

    'OrderID': 10688,
    'CustomerID': 'VAFFE',
    'OrderDate': '1997-10-01T00:00:00.000Z',
    'ShippedDate': '1997-10-07T00:00:00.000Z',
    'Freight': 299.09,
    'ShipName': 'Vaffeljernet',
    'ShipAddress': 'Smagsloget 45',
    'ShipCity': 'Århus',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10689,
    'CustomerID': 'BERGS',
    'OrderDate': '1997-10-01T00:00:00.000Z',
    'ShippedDate': '1997-10-07T00:00:00.000Z',
    'Freight': 13.42,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10690,
    'CustomerID': 'HANAR',
    'OrderDate': '1997-10-02T00:00:00.000Z',
    'ShippedDate': '1997-10-03T00:00:00.000Z',
    'Freight': 15.8,
    'ShipName': 'Hanari Carnes',
    'ShipAddress': 'Rua do Paço, 67',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10691,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-10-03T00:00:00.000Z',
    'ShippedDate': '1997-10-22T00:00:00.000Z',
    'Freight': 810.05,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10692,
    'CustomerID': 'ALFKI',
    'OrderDate': '1997-10-03T00:00:00.000Z',
    'ShippedDate': '1997-10-13T00:00:00.000Z',
    'Freight': 61.02,
    'ShipName': 'Alfred\ Futterkiste',
    'ShipAddress': 'Obere Str. 57',
    'ShipCity': 'Berlin',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10693,
    'CustomerID': 'WHITC',
    'OrderDate': '1997-10-06T00:00:00.000Z',
    'ShippedDate': '1997-10-10T00:00:00.000Z',
    'Freight': 139.34,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10694,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-10-06T00:00:00.000Z',
    'ShippedDate': '1997-10-09T00:00:00.000Z',
    'Freight': 398.36,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10695,
    'CustomerID': 'WILMK',
    'OrderDate': '1997-10-07T00:00:00.000Z',
    'ShippedDate': '1997-10-14T00:00:00.000Z',
    'Freight': 16.72,
    'ShipName': 'Wilman Kala',
    'ShipAddress': 'Keskuskatu 45',
    'ShipCity': 'Helsinki',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10696,
    'CustomerID': 'WHITC',
    'OrderDate': '1997-10-08T00:00:00.000Z',
    'ShippedDate': '1997-10-14T00:00:00.000Z',
    'Freight': 102.55,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10697,
    'CustomerID': 'LINOD',
    'OrderDate': '1997-10-08T00:00:00.000Z',
    'ShippedDate': '1997-10-14T00:00:00.000Z',
    'Freight': 45.52,
    'ShipName': 'LINO-Delicateses',

```

```

    'ShipAddress': 'Ave. 5 de Mayo Porlamar',
    'ShipCity': 'I. de Margarita',
    'ShipRegion': 'Nueva Esparta',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10698,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-10-09T00:00:00.000Z',
    'ShippedDate': '1997-10-17T00:00:00.000Z',
    'Freight': 272.47,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10699,
    'CustomerID': 'MORGK',
    'OrderDate': '1997-10-09T00:00:00.000Z',
    'ShippedDate': '1997-10-13T00:00:00.000Z',
    'Freight': 0.58,
    'ShipName': 'Morgenstern Gesundkost',
    'ShipAddress': 'Heerstr. 22',
    'ShipCity': 'Leipzig',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10700,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-10-10T00:00:00.000Z',
    'ShippedDate': '1997-10-16T00:00:00.000Z',
    'Freight': 65.1,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10701,
    'CustomerID': 'HUNGO',
    'OrderDate': '1997-10-13T00:00:00.000Z',
    'ShippedDate': '1997-10-15T00:00:00.000Z',
    'Freight': 220.31,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10702,
    'CustomerID': 'ALFKI',
    'OrderDate': '1997-10-13T00:00:00.000Z',

```

```

    'ShippedDate': '1997-10-21T00:00:00.000Z',
    'Freight': 23.94,
    'ShipName': 'Alfred\' Futterkiste',
    'ShipAddress': 'Obere Str. 57',
    'ShipCity': 'Berlin',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10703,
    'CustomerID': 'FOLKO',
    'OrderDate': '1997-10-14T00:00:00.000Z',
    'ShippedDate': '1997-10-20T00:00:00.000Z',
    'Freight': 152.3,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10704,
    'CustomerID': 'QUEEN',
    'OrderDate': '1997-10-14T00:00:00.000Z',
    'ShippedDate': '1997-11-07T00:00:00.000Z',
    'Freight': 4.78,
    'ShipName': 'Queen Cozinha',
    'ShipAddress': 'Alameda dos Canários, 891',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10705,
    'CustomerID': 'HILAA',
    'OrderDate': '1997-10-15T00:00:00.000Z',
    'ShippedDate': '1997-11-18T00:00:00.000Z',
    'Freight': 3.52,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10706,
    'CustomerID': 'OLDWO',
    'OrderDate': '1997-10-16T00:00:00.000Z',
    'ShippedDate': '1997-10-21T00:00:00.000Z',
    'Freight': 135.63,
    'ShipName': 'Old World Delicatessen',
    'ShipAddress': '2743 Bering St.',
    'ShipCity': 'Anchorage',
    'ShipRegion': 'AK',
    'ShipCountry': 'USA'
  },
  {

```

```

    'OrderID': 10707,
    'CustomerID': 'AROUT',
    'OrderDate': '1997-10-16T00:00:00.000Z',
    'ShippedDate': '1997-10-23T00:00:00.000Z',
    'Freight': 21.74,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10708,
    'CustomerID': 'THEBI',
    'OrderDate': '1997-10-17T00:00:00.000Z',
    'ShippedDate': '1997-11-05T00:00:00.000Z',
    'Freight': 2.96,
    'ShipName': 'The Big Cheese',
    'ShipAddress': '89 Jefferson Way Suite 2',
    'ShipCity': 'Portland',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10709,
    'CustomerID': 'GOURL',
    'OrderDate': '1997-10-17T00:00:00.000Z',
    'ShippedDate': '1997-11-20T00:00:00.000Z',
    'Freight': 210.8,
    'ShipName': 'Gourmet Lanchonetes',
    'ShipAddress': 'Av. Brasil, 442',
    'ShipCity': 'Campinas',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10710,
    'CustomerID': 'FRANS',
    'OrderDate': '1997-10-20T00:00:00.000Z',
    'ShippedDate': '1997-10-23T00:00:00.000Z',
    'Freight': 4.98,
    'ShipName': 'Franchi S.p.A.',
    'ShipAddress': 'Via Monte Bianco 34',
    'ShipCity': 'Torino',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10711,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-10-21T00:00:00.000Z',
    'ShippedDate': '1997-10-29T00:00:00.000Z',
    'Freight': 52.41,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',

```

```

        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10712,
        'CustomerID': 'HUNGO',
        'OrderDate': '1997-10-21T00:00:00.000Z',
        'ShippedDate': '1997-10-31T00:00:00.000Z',
        'Freight': 89.93,
        'ShipName': 'Hungry Owl All-Night Grocers',
        'ShipAddress': '8 Johnstown Road',
        'ShipCity': 'Cork',
        'ShipRegion': 'Co. Cork',
        'ShipCountry': 'Ireland'
    },
    {
        'OrderID': 10713,
        'CustomerID': 'SAVEA',
        'OrderDate': '1997-10-22T00:00:00.000Z',
        'ShippedDate': '1997-10-24T00:00:00.000Z',
        'Freight': 167.05,
        'ShipName': 'Save-a-lot Markets',
        'ShipAddress': '187 Suffolk Ln.',
        'ShipCity': 'Boise',
        'ShipRegion': 'ID',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10714,
        'CustomerID': 'SAVEA',
        'OrderDate': '1997-10-22T00:00:00.000Z',
        'ShippedDate': '1997-10-27T00:00:00.000Z',
        'Freight': 24.49,
        'ShipName': 'Save-a-lot Markets',
        'ShipAddress': '187 Suffolk Ln.',
        'ShipCity': 'Boise',
        'ShipRegion': 'ID',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10715,
        'CustomerID': 'BONAP',
        'OrderDate': '1997-10-23T00:00:00.000Z',
        'ShippedDate': '1997-10-29T00:00:00.000Z',
        'Freight': 63.2,
        'ShipName': 'Bon app',
        'ShipAddress': '12, rue des Bouchers',
        'ShipCity': 'Marseille',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10716,
        'CustomerID': 'RANCH',
        'OrderDate': '1997-10-24T00:00:00.000Z',
        'ShippedDate': '1997-10-27T00:00:00.000Z',
        'Freight': 22.57,
        'ShipName': 'Rancho grande',

```



```

        'ShipAddress': 'Av. del Libertador 900',
        'ShipCity': 'Buenos Aires',
        'ShipRegion': null,
        'ShipCountry': 'Argentina'
    },
    {
        'OrderID': 10717,
        'CustomerID': 'FRANK',
        'OrderDate': '1997-10-24T00:00:00.000Z',
        'ShippedDate': '1997-10-29T00:00:00.000Z',
        'Freight': 59.25,
        'ShipName': 'Frankenversand',
        'ShipAddress': 'Berliner Platz 43',
        'ShipCity': 'München',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10718,
        'CustomerID': 'KOENE',
        'OrderDate': '1997-10-27T00:00:00.000Z',
        'ShippedDate': '1997-10-29T00:00:00.000Z',
        'Freight': 170.88,
        'ShipName': 'Königlich Essen',
        'ShipAddress': 'Maubelstr. 90',
        'ShipCity': 'Brandenburg',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10719,
        'CustomerID': 'LETSS',
        'OrderDate': '1997-10-27T00:00:00.000Z',
        'ShippedDate': '1997-11-05T00:00:00.000Z',
        'Freight': 51.44,
        'ShipName': 'Let\' Stop N Shop',
        'ShipAddress': '87 Polk St. Suite 5',
        'ShipCity': 'San Francisco',
        'ShipRegion': 'CA',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10720,
        'CustomerID': 'QUEDE',
        'OrderDate': '1997-10-28T00:00:00.000Z',
        'ShippedDate': '1997-11-05T00:00:00.000Z',
        'Freight': 9.53,
        'ShipName': 'Que Delícia',
        'ShipAddress': 'Rua da Panificadora, 12',
        'ShipCity': 'Rio de Janeiro',
        'ShipRegion': 'RJ',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10721,
        'CustomerID': 'QUICK',
        'OrderDate': '1997-10-29T00:00:00.000Z',

```

```

    'ShippedDate': '1997-10-31T00:00:00.000Z',
    'Freight': 48.92,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10722,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-10-29T00:00:00.000Z',
    'ShippedDate': '1997-11-04T00:00:00.000Z',
    'Freight': 74.58,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10723,
    'CustomerID': 'WHITC',
    'OrderDate': '1997-10-30T00:00:00.000Z',
    'ShippedDate': '1997-11-25T00:00:00.000Z',
    'Freight': 21.72,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10724,
    'CustomerID': 'MEREP',
    'OrderDate': '1997-10-30T00:00:00.000Z',
    'ShippedDate': '1997-11-05T00:00:00.000Z',
    'Freight': 57.75,
    'ShipName': 'Mère Paillarde',
    'ShipAddress': '43 rue St. Laurent',
    'ShipCity': 'Montréal',
    'ShipRegion': 'Québec',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10725,
    'CustomerID': 'FAMIA',
    'OrderDate': '1997-10-31T00:00:00.000Z',
    'ShippedDate': '1997-11-05T00:00:00.000Z',
    'Freight': 10.83,
    'ShipName': 'Familia Arquibaldo',
    'ShipAddress': 'Rua Orós, 92',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {

```

```

    'OrderID': 10726,
    'CustomerID': 'EASTC',
    'OrderDate': '1997-11-03T00:00:00.000Z',
    'ShippedDate': '1997-12-05T00:00:00.000Z',
    'Freight': 16.56,
    'ShipName': 'Eastern Connection',
    'ShipAddress': '35 King George',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10727,
    'CustomerID': 'REGGC',
    'OrderDate': '1997-11-03T00:00:00.000Z',
    'ShippedDate': '1997-12-05T00:00:00.000Z',
    'Freight': 89.9,
    'ShipName': 'Reggiani Caseifici',
    'ShipAddress': 'Strada Provinciale 124',
    'ShipCity': 'Reggio Emilia',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10728,
    'CustomerID': 'QUEEN',
    'OrderDate': '1997-11-04T00:00:00.000Z',
    'ShippedDate': '1997-11-11T00:00:00.000Z',
    'Freight': 58.33,
    'ShipName': 'Queen Cozinha',
    'ShipAddress': 'Alameda dos Canários, 891',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10729,
    'CustomerID': 'LINOD',
    'OrderDate': '1997-11-04T00:00:00.000Z',
    'ShippedDate': '1997-11-14T00:00:00.000Z',
    'Freight': 141.06,
    'ShipName': 'LINO-Delicateses',
    'ShipAddress': 'Ave. 5 de Mayo Porlamar',
    'ShipCity': 'I. de Margarita',
    'ShipRegion': 'Nueva Esparta',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10730,
    'CustomerID': 'BONAP',
    'OrderDate': '1997-11-05T00:00:00.000Z',
    'ShippedDate': '1997-11-14T00:00:00.000Z',
    'Freight': 20.12,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,

```

```

    'ShipCountry': 'France'
  },
  {
    'OrderID': 10731,
    'CustomerID': 'CHOPS',
    'OrderDate': '1997-11-06T00:00:00.000Z',
    'ShippedDate': '1997-11-14T00:00:00.000Z',
    'Freight': 96.65,
    'ShipName': 'Chop-suey Chinese',
    'ShipAddress': 'Hauptstr. 31',
    'ShipCity': 'Bern',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10732,
    'CustomerID': 'BONAP',
    'OrderDate': '1997-11-06T00:00:00.000Z',
    'ShippedDate': '1997-11-07T00:00:00.000Z',
    'Freight': 16.97,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10733,
    'CustomerID': 'BERGS',
    'OrderDate': '1997-11-07T00:00:00.000Z',
    'ShippedDate': '1997-11-10T00:00:00.000Z',
    'Freight': 110.11,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10734,
    'CustomerID': 'GOURL',
    'OrderDate': '1997-11-07T00:00:00.000Z',
    'ShippedDate': '1997-11-12T00:00:00.000Z',
    'Freight': 1.63,
    'ShipName': 'Gourmet Lanchonetes',
    'ShipAddress': 'Av. Brasil, 442',
    'ShipCity': 'Campinas',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10735,
    'CustomerID': 'LETSS',
    'OrderDate': '1997-11-10T00:00:00.000Z',
    'ShippedDate': '1997-11-21T00:00:00.000Z',
    'Freight': 45.97,
    'ShipName': 'Let\ Stop N Shop',

```

```

        'ShipAddress': '87 Polk St. Suite 5',
        'ShipCity': 'San Francisco',
        'ShipRegion': 'CA',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10736,
        'CustomerID': 'HUNGO',
        'OrderDate': '1997-11-11T00:00:00.000Z',
        'ShippedDate': '1997-11-21T00:00:00.000Z',
        'Freight': 44.1,
        'ShipName': 'Hungry Owl All-Night Grocers',
        'ShipAddress': '8 Johnstown Road',
        'ShipCity': 'Cork',
        'ShipRegion': 'Co. Cork',
        'ShipCountry': 'Ireland'
    },
    {
        'OrderID': 10737,
        'CustomerID': 'VINET',
        'OrderDate': '1997-11-11T00:00:00.000Z',
        'ShippedDate': '1997-11-18T00:00:00.000Z',
        'Freight': 7.79,
        'ShipName': 'Vins et alcools Chevalier',
        'ShipAddress': '59 rue de l\'Abbaye',
        'ShipCity': 'Reims',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10738,
        'CustomerID': 'SPECB',
        'OrderDate': '1997-11-12T00:00:00.000Z',
        'ShippedDate': '1997-11-18T00:00:00.000Z',
        'Freight': 2.91,
        'ShipName': 'Spécialités du monde',
        'ShipAddress': '25, rue Lauriston',
        'ShipCity': 'Paris',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10739,
        'CustomerID': 'VINET',
        'OrderDate': '1997-11-12T00:00:00.000Z',
        'ShippedDate': '1997-11-17T00:00:00.000Z',
        'Freight': 11.08,
        'ShipName': 'Vins et alcools Chevalier',
        'ShipAddress': '59 rue de l\'Abbaye',
        'ShipCity': 'Reims',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10740,
        'CustomerID': 'WHITC',
        'OrderDate': '1997-11-13T00:00:00.000Z',

```

```

    'ShippedDate': '1997-11-25T00:00:00.000Z',
    'Freight': 81.88,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10741,
    'CustomerID': 'AROUT',
    'OrderDate': '1997-11-14T00:00:00.000Z',
    'ShippedDate': '1997-11-18T00:00:00.000Z',
    'Freight': 10.96,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10742,
    'CustomerID': 'BOTTM',
    'OrderDate': '1997-11-14T00:00:00.000Z',
    'ShippedDate': '1997-11-18T00:00:00.000Z',
    'Freight': 243.73,
    'ShipName': 'Bottom-Dollar Markets',
    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10743,
    'CustomerID': 'AROUT',
    'OrderDate': '1997-11-17T00:00:00.000Z',
    'ShippedDate': '1997-11-21T00:00:00.000Z',
    'Freight': 23.72,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10744,
    'CustomerID': 'VAFFE',
    'OrderDate': '1997-11-17T00:00:00.000Z',
    'ShippedDate': '1997-11-24T00:00:00.000Z',
    'Freight': 69.19,
    'ShipName': 'Vaffeljernet',
    'ShipAddress': 'Smagsloget 45',
    'ShipCity': 'Århus',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {

```

```

    'OrderID': 10745,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-11-18T00:00:00.000Z',
    'ShippedDate': '1997-11-27T00:00:00.000Z',
    'Freight': 3.52,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10746,
    'CustomerID': 'CHOPS',
    'OrderDate': '1997-11-19T00:00:00.000Z',
    'ShippedDate': '1997-11-21T00:00:00.000Z',
    'Freight': 31.43,
    'ShipName': 'Chop-suey Chinese',
    'ShipAddress': 'Hauptstr. 31',
    'ShipCity': 'Bern',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10747,
    'CustomerID': 'PICCO',
    'OrderDate': '1997-11-19T00:00:00.000Z',
    'ShippedDate': '1997-11-26T00:00:00.000Z',
    'Freight': 117.33,
    'ShipName': 'Piccolo und mehr',
    'ShipAddress': 'Geislweg 14',
    'ShipCity': 'Salzburg',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10748,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-11-20T00:00:00.000Z',
    'ShippedDate': '1997-11-28T00:00:00.000Z',
    'Freight': 232.55,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10749,
    'CustomerID': 'ISLAT',
    'OrderDate': '1997-11-20T00:00:00.000Z',
    'ShippedDate': '1997-12-19T00:00:00.000Z',
    'Freight': 61.53,
    'ShipName': 'Island Trading',
    'ShipAddress': 'Garden House Crowther Way',
    'ShipCity': 'Cowes',
    'ShipRegion': 'Isle of Wight',

```

```

    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10750,
    'CustomerID': 'WARTH',
    'OrderDate': '1997-11-21T00:00:00.000Z',
    'ShippedDate': '1997-11-24T00:00:00.000Z',
    'Freight': 79.3,
    'ShipName': 'Wartian Herkku',
    'ShipAddress': 'Torikatu 38',
    'ShipCity': 'Oulu',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10751,
    'CustomerID': 'RICSU',
    'OrderDate': '1997-11-24T00:00:00.000Z',
    'ShippedDate': '1997-12-03T00:00:00.000Z',
    'Freight': 130.79,
    'ShipName': 'Richter Supermarkt',
    'ShipAddress': 'Starenweg 5',
    'ShipCity': 'Genève',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10752,
    'CustomerID': 'NORTS',
    'OrderDate': '1997-11-24T00:00:00.000Z',
    'ShippedDate': '1997-11-28T00:00:00.000Z',
    'Freight': 1.39,
    'ShipName': 'North/South',
    'ShipAddress': 'South House 300 Queensbridge',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10753,
    'CustomerID': 'FRANS',
    'OrderDate': '1997-11-25T00:00:00.000Z',
    'ShippedDate': '1997-11-27T00:00:00.000Z',
    'Freight': 7.7,
    'ShipName': 'Franchi S.p.A.',
    'ShipAddress': 'Via Monte Bianco 34',
    'ShipCity': 'Torino',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10754,
    'CustomerID': 'MAGAA',
    'OrderDate': '1997-11-25T00:00:00.000Z',
    'ShippedDate': '1997-11-27T00:00:00.000Z',
    'Freight': 2.38,
    'ShipName': 'Magazzini Alimentari Riuniti',

```



```

    'ShipAddress': 'Via Ludovico il Moro 22',
    'ShipCity': 'Bergamo',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10755,
    'CustomerID': 'BONAP',
    'OrderDate': '1997-11-26T00:00:00.000Z',
    'ShippedDate': '1997-11-28T00:00:00.000Z',
    'Freight': 16.71,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10756,
    'CustomerID': 'SPLIR',
    'OrderDate': '1997-11-27T00:00:00.000Z',
    'ShippedDate': '1997-12-02T00:00:00.000Z',
    'Freight': 73.21,
    'ShipName': 'Split Rail Beer & Ale',
    'ShipAddress': 'P.O. Box 555',
    'ShipCity': 'Lander',
    'ShipRegion': 'WY',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10757,
    'CustomerID': 'SAVEA',
    'OrderDate': '1997-11-27T00:00:00.000Z',
    'ShippedDate': '1997-12-15T00:00:00.000Z',
    'Freight': 8.19,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10758,
    'CustomerID': 'RICSU',
    'OrderDate': '1997-11-28T00:00:00.000Z',
    'ShippedDate': '1997-12-04T00:00:00.000Z',
    'Freight': 138.17,
    'ShipName': 'Richter Supermarkt',
    'ShipAddress': 'Starenweg 5',
    'ShipCity': 'Genève',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10759,
    'CustomerID': 'ANATR',
    'OrderDate': '1997-11-28T00:00:00.000Z',

```

```

    'ShippedDate': '1997-12-12T00:00:00.000Z',
    'Freight': 11.99,
    'ShipName': 'Ana Trujillo Emparedados y helados',
    'ShipAddress': 'Avda. de la Constitución 2222',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10760,
    'CustomerID': 'MAISD',
    'OrderDate': '1997-12-01T00:00:00.000Z',
    'ShippedDate': '1997-12-10T00:00:00.000Z',
    'Freight': 155.64,
    'ShipName': 'Maison Dewey',
    'ShipAddress': 'Rue Joseph-Bens 532',
    'ShipCity': 'Bruxelles',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10761,
    'CustomerID': 'RATTC',
    'OrderDate': '1997-12-02T00:00:00.000Z',
    'ShippedDate': '1997-12-08T00:00:00.000Z',
    'Freight': 18.66,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10762,
    'CustomerID': 'FOLKO',
    'OrderDate': '1997-12-02T00:00:00.000Z',
    'ShippedDate': '1997-12-09T00:00:00.000Z',
    'Freight': 328.74,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10763,
    'CustomerID': 'FOLIG',
    'OrderDate': '1997-12-03T00:00:00.000Z',
    'ShippedDate': '1997-12-08T00:00:00.000Z',
    'Freight': 37.35,
    'ShipName': 'Folies gourmandes',
    'ShipAddress': '184, chaussée de Tournai',
    'ShipCity': 'Lille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {

```

```

    'OrderID': 10764,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-12-03T00:00:00.000Z',
    'ShippedDate': '1997-12-08T00:00:00.000Z',
    'Freight': 145.45,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10765,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-12-04T00:00:00.000Z',
    'ShippedDate': '1997-12-09T00:00:00.000Z',
    'Freight': 42.74,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10766,
    'CustomerID': 'OTTIK',
    'OrderDate': '1997-12-05T00:00:00.000Z',
    'ShippedDate': '1997-12-09T00:00:00.000Z',
    'Freight': 157.55,
    'ShipName': 'Ottilies Käseladen',
    'ShipAddress': 'Mehrheimerstr. 369',
    'ShipCity': 'Köln',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10767,
    'CustomerID': 'SUPRD',
    'OrderDate': '1997-12-05T00:00:00.000Z',
    'ShippedDate': '1997-12-15T00:00:00.000Z',
    'Freight': 1.59,
    'ShipName': 'Suprêmes délices',
    'ShipAddress': 'Boulevard Tirou, 255',
    'ShipCity': 'Charleroi',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10768,
    'CustomerID': 'AROUT',
    'OrderDate': '1997-12-08T00:00:00.000Z',
    'ShippedDate': '1997-12-15T00:00:00.000Z',
    'Freight': 146.32,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',
  }

```

```

    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10769,
    'CustomerID': 'VAFFE',
    'OrderDate': '1997-12-08T00:00:00.000Z',
    'ShippedDate': '1997-12-12T00:00:00.000Z',
    'Freight': 65.06,
    'ShipName': 'Vaffeljernet',
    'ShipAddress': 'Smagsloget 45',
    'ShipCity': 'Århus',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10770,
    'CustomerID': 'HANAR',
    'OrderDate': '1997-12-09T00:00:00.000Z',
    'ShippedDate': '1997-12-17T00:00:00.000Z',
    'Freight': 5.32,
    'ShipName': 'Hanari Carnes',
    'ShipAddress': 'Rua do Paço, 67',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10771,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-12-10T00:00:00.000Z',
    'ShippedDate': '1998-01-02T00:00:00.000Z',
    'Freight': 11.19,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10772,
    'CustomerID': 'LEHMS',
    'OrderDate': '1997-12-10T00:00:00.000Z',
    'ShippedDate': '1997-12-19T00:00:00.000Z',
    'Freight': 91.28,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10773,
    'CustomerID': 'ERNSH',
    'OrderDate': '1997-12-11T00:00:00.000Z',
    'ShippedDate': '1997-12-16T00:00:00.000Z',
    'Freight': 96.43,
    'ShipName': 'Ernst Handel',

```

```

        'ShipAddress': 'Kirchgasse 6',
        'ShipCity': 'Graz',
        'ShipRegion': null,
        'ShipCountry': 'Austria'
    },
    {
        'OrderID': 10774,
        'CustomerID': 'FOLKO',
        'OrderDate': '1997-12-11T00:00:00.000Z',
        'ShippedDate': '1997-12-12T00:00:00.000Z',
        'Freight': 48.2,
        'ShipName': 'Folk och fä HB',
        'ShipAddress': 'Åkergatan 24',
        'ShipCity': 'Bräcke',
        'ShipRegion': null,
        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 10775,
        'CustomerID': 'THECR',
        'OrderDate': '1997-12-12T00:00:00.000Z',
        'ShippedDate': '1997-12-26T00:00:00.000Z',
        'Freight': 20.25,
        'ShipName': 'The Cracker Box',
        'ShipAddress': '55 Grizzly Peak Rd.',
        'ShipCity': 'Butte',
        'ShipRegion': 'MT',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10776,
        'CustomerID': 'ERNSH',
        'OrderDate': '1997-12-15T00:00:00.000Z',
        'ShippedDate': '1997-12-18T00:00:00.000Z',
        'Freight': 351.53,
        'ShipName': 'Ernst Handel',
        'ShipAddress': 'Kirchgasse 6',
        'ShipCity': 'Graz',
        'ShipRegion': null,
        'ShipCountry': 'Austria'
    },
    {
        'OrderID': 10777,
        'CustomerID': 'GOURL',
        'OrderDate': '1997-12-15T00:00:00.000Z',
        'ShippedDate': '1998-01-21T00:00:00.000Z',
        'Freight': 3.01,
        'ShipName': 'Gourmet Lanchonetes',
        'ShipAddress': 'Av. Brasil, 442',
        'ShipCity': 'Campinas',
        'ShipRegion': 'SP',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10778,
        'CustomerID': 'BERGS',
        'OrderDate': '1997-12-16T00:00:00.000Z',

```

```

    'ShippedDate': '1997-12-24T00:00:00.000Z',
    'Freight': 6.79,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10779,
    'CustomerID': 'MORGK',
    'OrderDate': '1997-12-16T00:00:00.000Z',
    'ShippedDate': '1998-01-14T00:00:00.000Z',
    'Freight': 58.13,
    'ShipName': 'Morgenstern Gesundkost',
    'ShipAddress': 'Heerstr. 22',
    'ShipCity': 'Leipzig',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10780,
    'CustomerID': 'LILAS',
    'OrderDate': '1997-12-16T00:00:00.000Z',
    'ShippedDate': '1997-12-25T00:00:00.000Z',
    'Freight': 42.13,
    'ShipName': 'LILA-Supermercado',
    'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10781,
    'CustomerID': 'WARTH',
    'OrderDate': '1997-12-17T00:00:00.000Z',
    'ShippedDate': '1997-12-19T00:00:00.000Z',
    'Freight': 73.16,
    'ShipName': 'Wartian Herkku',
    'ShipAddress': 'Torikatu 38',
    'ShipCity': 'Oulu',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10782,
    'CustomerID': 'CACTU',
    'OrderDate': '1997-12-17T00:00:00.000Z',
    'ShippedDate': '1997-12-22T00:00:00.000Z',
    'Freight': 1.1,
    'ShipName': 'Cactus Comidas para llevar',
    'ShipAddress': 'Cerrito 333',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  },
  {

```

```

    'OrderID': 10783,
    'CustomerID': 'HANAR',
    'OrderDate': '1997-12-18T00:00:00.000Z',
    'ShippedDate': '1997-12-19T00:00:00.000Z',
    'Freight': 124.98,
    'ShipName': 'Hanari Carnes',
    'ShipAddress': 'Rua do Paço, 67',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10784,
    'CustomerID': 'MAGAA',
    'OrderDate': '1997-12-18T00:00:00.000Z',
    'ShippedDate': '1997-12-22T00:00:00.000Z',
    'Freight': 70.09,
    'ShipName': 'Magazzini Alimentari Riuniti',
    'ShipAddress': 'Via Ludovico il Moro 22',
    'ShipCity': 'Bergamo',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10785,
    'CustomerID': 'GROSR',
    'OrderDate': '1997-12-18T00:00:00.000Z',
    'ShippedDate': '1997-12-24T00:00:00.000Z',
    'Freight': 1.51,
    'ShipName': 'GROSELLA-Restaurante',
    'ShipAddress': '5ª Ave. Los Palos Grandes',
    'ShipCity': 'Caracas',
    'ShipRegion': 'DF',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10786,
    'CustomerID': 'QUEEN',
    'OrderDate': '1997-12-19T00:00:00.000Z',
    'ShippedDate': '1997-12-23T00:00:00.000Z',
    'Freight': 110.87,
    'ShipName': 'Queen Cozinha',
    'ShipAddress': 'Alameda dos Canários, 891',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10787,
    'CustomerID': 'LAMAI',
    'OrderDate': '1997-12-19T00:00:00.000Z',
    'ShippedDate': '1997-12-26T00:00:00.000Z',
    'Freight': 249.93,
    'ShipName': 'La maison d\'Asie',
    'ShipAddress': '1 rue Alsace-Lorraine',
    'ShipCity': 'Toulouse',
    'ShipRegion': null,

```

```

    'ShipCountry': 'France'
  },
  {
    'OrderID': 10788,
    'CustomerID': 'QUICK',
    'OrderDate': '1997-12-22T00:00:00.000Z',
    'ShippedDate': '1998-01-19T00:00:00.000Z',
    'Freight': 42.7,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10789,
    'CustomerID': 'FOLIG',
    'OrderDate': '1997-12-22T00:00:00.000Z',
    'ShippedDate': '1997-12-31T00:00:00.000Z',
    'Freight': 100.6,
    'ShipName': 'Folies gourmandes',
    'ShipAddress': '184, chaussée de Tournai',
    'ShipCity': 'Lille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10790,
    'CustomerID': 'GOURL',
    'OrderDate': '1997-12-22T00:00:00.000Z',
    'ShippedDate': '1997-12-26T00:00:00.000Z',
    'Freight': 28.23,
    'ShipName': 'Gourmet Lanchonetes',
    'ShipAddress': 'Av. Brasil, 442',
    'ShipCity': 'Campinas',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10791,
    'CustomerID': 'FRANK',
    'OrderDate': '1997-12-23T00:00:00.000Z',
    'ShippedDate': '1998-01-01T00:00:00.000Z',
    'Freight': 16.85,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10792,
    'CustomerID': 'WOLZA',
    'OrderDate': '1997-12-23T00:00:00.000Z',
    'ShippedDate': '1997-12-31T00:00:00.000Z',
    'Freight': 23.79,
    'ShipName': 'Wolski Zajazd',

```



```

        'ShipAddress': 'ul. Filtrowa 68',
        'ShipCity': 'Warszawa',
        'ShipRegion': null,
        'ShipCountry': 'Poland'
    },
    {
        'OrderID': 10793,
        'CustomerID': 'AROUT',
        'OrderDate': '1997-12-24T00:00:00.000Z',
        'ShippedDate': '1998-01-08T00:00:00.000Z',
        'Freight': 4.52,
        'ShipName': 'Around the Horn',
        'ShipAddress': 'Brook Farm Stratford St. Mary',
        'ShipCity': 'Colchester',
        'ShipRegion': 'Essex',
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10794,
        'CustomerID': 'QUEDE',
        'OrderDate': '1997-12-24T00:00:00.000Z',
        'ShippedDate': '1998-01-02T00:00:00.000Z',
        'Freight': 21.49,
        'ShipName': 'Que Delícia',
        'ShipAddress': 'Rua da Panificadora, 12',
        'ShipCity': 'Rio de Janeiro',
        'ShipRegion': 'RJ',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10795,
        'CustomerID': 'ERNSH',
        'OrderDate': '1997-12-24T00:00:00.000Z',
        'ShippedDate': '1998-01-20T00:00:00.000Z',
        'Freight': 126.66,
        'ShipName': 'Ernst Handel',
        'ShipAddress': 'Kirchgasse 6',
        'ShipCity': 'Graz',
        'ShipRegion': null,
        'ShipCountry': 'Austria'
    },
    {
        'OrderID': 10796,
        'CustomerID': 'HILAA',
        'OrderDate': '1997-12-25T00:00:00.000Z',
        'ShippedDate': '1998-01-14T00:00:00.000Z',
        'Freight': 26.52,
        'ShipName': 'HILARION-Abastos',
        'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
        'ShipCity': 'San Cristóbal',
        'ShipRegion': 'Táchira',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 10797,
        'CustomerID': 'DRACD',
        'OrderDate': '1997-12-25T00:00:00.000Z',

```

```

        'ShippedDate': '1998-01-05T00:00:00.000Z',
        'Freight': 33.35,
        'ShipName': 'Drachenblut Delikatessen',
        'ShipAddress': 'Walserweg 21',
        'ShipCity': 'Aachen',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10798,
        'CustomerID': 'ISLAT',
        'OrderDate': '1997-12-26T00:00:00.000Z',
        'ShippedDate': '1998-01-05T00:00:00.000Z',
        'Freight': 2.33,
        'ShipName': 'Island Trading',
        'ShipAddress': 'Garden House Crowther Way',
        'ShipCity': 'Cowes',
        'ShipRegion': 'Isle of Wight',
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10799,
        'CustomerID': 'KOENE',
        'OrderDate': '1997-12-26T00:00:00.000Z',
        'ShippedDate': '1998-01-05T00:00:00.000Z',
        'Freight': 30.76,
        'ShipName': 'Königlich Essen',
        'ShipAddress': 'Maubelstr. 90',
        'ShipCity': 'Brandenburg',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 10800,
        'CustomerID': 'SEVES',
        'OrderDate': '1997-12-26T00:00:00.000Z',
        'ShippedDate': '1998-01-05T00:00:00.000Z',
        'Freight': 137.44,
        'ShipName': 'Seven Seas Imports',
        'ShipAddress': '90 Wadhurst Rd.',
        'ShipCity': 'London',
        'ShipRegion': null,
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10801,
        'CustomerID': 'BOLID',
        'OrderDate': '1997-12-29T00:00:00.000Z',
        'ShippedDate': '1997-12-31T00:00:00.000Z',
        'Freight': 97.09,
        'ShipName': 'Bólido Comidas preparadas',
        'ShipAddress': 'C/ Araquil, 67',
        'ShipCity': 'Madrid',
        'ShipRegion': null,
        'ShipCountry': 'Spain'
    },
    {

```

```

    'OrderID': 10802,
    'CustomerID': 'SIMOB',
    'OrderDate': '1997-12-29T00:00:00.000Z',
    'ShippedDate': '1998-01-02T00:00:00.000Z',
    'Freight': 257.26,
    'ShipName': 'Simons bistro',
    'ShipAddress': 'Vinbæltet 34',
    'ShipCity': 'Kobenhavn',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10803,
    'CustomerID': 'WELLI',
    'OrderDate': '1997-12-30T00:00:00.000Z',
    'ShippedDate': '1998-01-06T00:00:00.000Z',
    'Freight': 55.23,
    'ShipName': 'Wellington Importadora',
    'ShipAddress': 'Rua do Mercado, 12',
    'ShipCity': 'Resende',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10804,
    'CustomerID': 'SEVES',
    'OrderDate': '1997-12-30T00:00:00.000Z',
    'ShippedDate': '1998-01-07T00:00:00.000Z',
    'Freight': 27.33,
    'ShipName': 'Seven Seas Imports',
    'ShipAddress': '90 Wadhurst Rd.',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10805,
    'CustomerID': 'THEBI',
    'OrderDate': '1997-12-30T00:00:00.000Z',
    'ShippedDate': '1998-01-09T00:00:00.000Z',
    'Freight': 237.34,
    'ShipName': 'The Big Cheese',
    'ShipAddress': '89 Jefferson Way Suite 2',
    'ShipCity': 'Portland',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10806,
    'CustomerID': 'VICTE',
    'OrderDate': '1997-12-31T00:00:00.000Z',
    'ShippedDate': '1998-01-05T00:00:00.000Z',
    'Freight': 22.11,
    'ShipName': 'Victuailles en stock',
    'ShipAddress': '2, rue du Commerce',
    'ShipCity': 'Lyon',
    'ShipRegion': null,

```

```

    'ShipCountry': 'France'
  },
  {
    'OrderID': 10807,
    'CustomerID': 'FRANS',
    'OrderDate': '1997-12-31T00:00:00.000Z',
    'ShippedDate': '1998-01-30T00:00:00.000Z',
    'Freight': 1.36,
    'ShipName': 'Franchi S.p.A.',
    'ShipAddress': 'Via Monte Bianco 34',
    'ShipCity': 'Torino',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10808,
    'CustomerID': 'OLDWO',
    'OrderDate': '1998-01-01T00:00:00.000Z',
    'ShippedDate': '1998-01-09T00:00:00.000Z',
    'Freight': 45.53,
    'ShipName': 'Old World Delicatessen',
    'ShipAddress': '2743 Bering St.',
    'ShipCity': 'Anchorage',
    'ShipRegion': 'AK',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10809,
    'CustomerID': 'WELLI',
    'OrderDate': '1998-01-01T00:00:00.000Z',
    'ShippedDate': '1998-01-07T00:00:00.000Z',
    'Freight': 4.87,
    'ShipName': 'Wellington Importadora',
    'ShipAddress': 'Rua do Mercado, 12',
    'ShipCity': 'Resende',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10810,
    'CustomerID': 'LAUGB',
    'OrderDate': '1998-01-01T00:00:00.000Z',
    'ShippedDate': '1998-01-07T00:00:00.000Z',
    'Freight': 4.33,
    'ShipName': 'Laughing Bacchus Wine Cellars',
    'ShipAddress': '2319 Elm St.',
    'ShipCity': 'Vancouver',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10811,
    'CustomerID': 'LINOD',
    'OrderDate': '1998-01-02T00:00:00.000Z',
    'ShippedDate': '1998-01-08T00:00:00.000Z',
    'Freight': 31.22,
    'ShipName': 'LINO-Delicateses',

```

```

    'ShipAddress': 'Ave. 5 de Mayo Porlamar',
    'ShipCity': 'I. de Margarita',
    'ShipRegion': 'Nueva Esparta',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10812,
    'CustomerID': 'REGGC',
    'OrderDate': '1998-01-02T00:00:00.000Z',
    'ShippedDate': '1998-01-12T00:00:00.000Z',
    'Freight': 59.78,
    'ShipName': 'Reggiani Caseifici',
    'ShipAddress': 'Strada Provinciale 124',
    'ShipCity': 'Reggio Emilia',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10813,
    'CustomerID': 'RICAR',
    'OrderDate': '1998-01-05T00:00:00.000Z',
    'ShippedDate': '1998-01-09T00:00:00.000Z',
    'Freight': 47.38,
    'ShipName': 'Ricardo Adocicados',
    'ShipAddress': 'Av. Copacabana, 267',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10814,
    'CustomerID': 'VICTE',
    'OrderDate': '1998-01-05T00:00:00.000Z',
    'ShippedDate': '1998-01-14T00:00:00.000Z',
    'Freight': 130.94,
    'ShipName': 'Victuailles en stock',
    'ShipAddress': '2, rue du Commerce',
    'ShipCity': 'Lyon',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10815,
    'CustomerID': 'SAVEA',
    'OrderDate': '1998-01-05T00:00:00.000Z',
    'ShippedDate': '1998-01-14T00:00:00.000Z',
    'Freight': 14.62,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10816,
    'CustomerID': 'GREAL',
    'OrderDate': '1998-01-06T00:00:00.000Z',

```

```

    'ShippedDate': '1998-02-04T00:00:00.000Z',
    'Freight': 719.78,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10817,
    'CustomerID': 'KOENE',
    'OrderDate': '1998-01-06T00:00:00.000Z',
    'ShippedDate': '1998-01-13T00:00:00.000Z',
    'Freight': 306.07,
    'ShipName': 'Königlich Essen',
    'ShipAddress': 'Maubelstr. 90',
    'ShipCity': 'Brandenburg',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10818,
    'CustomerID': 'MAGAA',
    'OrderDate': '1998-01-07T00:00:00.000Z',
    'ShippedDate': '1998-01-12T00:00:00.000Z',
    'Freight': 65.48,
    'ShipName': 'Magazzini Alimentari Riuniti',
    'ShipAddress': 'Via Ludovico il Moro 22',
    'ShipCity': 'Bergamo',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10819,
    'CustomerID': 'CACTU',
    'OrderDate': '1998-01-07T00:00:00.000Z',
    'ShippedDate': '1998-01-16T00:00:00.000Z',
    'Freight': 19.76,
    'ShipName': 'Cactus Comidas para llevar',
    'ShipAddress': 'Cerrito 333',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 10820,
    'CustomerID': 'RATTC',
    'OrderDate': '1998-01-07T00:00:00.000Z',
    'ShippedDate': '1998-01-13T00:00:00.000Z',
    'Freight': 37.52,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {

```

```

    'OrderID': 10821,
    'CustomerID': 'SPLIR',
    'OrderDate': '1998-01-08T00:00:00.000Z',
    'ShippedDate': '1998-01-15T00:00:00.000Z',
    'Freight': 36.68,
    'ShipName': 'Split Rail Beer & Ale',
    'ShipAddress': 'P.O. Box 555',
    'ShipCity': 'Lander',
    'ShipRegion': 'WY',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10822,
    'CustomerID': 'TRAIH',
    'OrderDate': '1998-01-08T00:00:00.000Z',
    'ShippedDate': '1998-01-16T00:00:00.000Z',
    'Freight': 7,
    'ShipName': 'Trail\ Head Gourmet Provisioners',
    'ShipAddress': '722 DaVinci Blvd.',
    'ShipCity': 'Kirkland',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10823,
    'CustomerID': 'LILAS',
    'OrderDate': '1998-01-09T00:00:00.000Z',
    'ShippedDate': '1998-01-13T00:00:00.000Z',
    'Freight': 163.97,
    'ShipName': 'LILA-Supermercado',
    'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10824,
    'CustomerID': 'FOLKO',
    'OrderDate': '1998-01-09T00:00:00.000Z',
    'ShippedDate': '1998-01-30T00:00:00.000Z',
    'Freight': 1.23,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10825,
    'CustomerID': 'DRACD',
    'OrderDate': '1998-01-09T00:00:00.000Z',
    'ShippedDate': '1998-01-14T00:00:00.000Z',
    'Freight': 79.25,
    'ShipName': 'Drachenblut Delikatessen',
    'ShipAddress': 'Walserweg 21',
    'ShipCity': 'Aachen',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10826,
    'CustomerID': 'BLONP',
    'OrderDate': '1998-01-12T00:00:00.000Z',
    'ShippedDate': '1998-02-06T00:00:00.000Z',
    'Freight': 7.09,
    'ShipName': 'Blondel père et fils',
    'ShipAddress': '24, place Kléber',
    'ShipCity': 'Strasbourg',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10827,
    'CustomerID': 'BONAP',
    'OrderDate': '1998-01-12T00:00:00.000Z',
    'ShippedDate': '1998-02-06T00:00:00.000Z',
    'Freight': 63.54,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10828,
    'CustomerID': 'RANCH',
    'OrderDate': '1998-01-13T00:00:00.000Z',
    'ShippedDate': '1998-02-04T00:00:00.000Z',
    'Freight': 90.85,
    'ShipName': 'Rancho grande',
    'ShipAddress': 'Av. del Libertador 900',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 10829,
    'CustomerID': 'ISLAT',
    'OrderDate': '1998-01-13T00:00:00.000Z',
    'ShippedDate': '1998-01-23T00:00:00.000Z',
    'Freight': 154.72,
    'ShipName': 'Island Trading',
    'ShipAddress': 'Garden House Crowther Way',
    'ShipCity': 'Cowes',
    'ShipRegion': 'Isle of Wight',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10830,
    'CustomerID': 'TRADH',
    'OrderDate': '1998-01-13T00:00:00.000Z',
    'ShippedDate': '1998-01-21T00:00:00.000Z',
    'Freight': 81.83,
    'ShipName': 'Tradição Hipermercados',

```



```

    'ShipAddress': 'Av. Inês de Castro, 414',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10831,
    'CustomerID': 'SANTG',
    'OrderDate': '1998-01-14T00:00:00.000Z',
    'ShippedDate': '1998-01-23T00:00:00.000Z',
    'Freight': 72.19,
    'ShipName': 'Santé Gourmet',
    'ShipAddress': 'Erling Skakkes gate 78',
    'ShipCity': 'Stavern',
    'ShipRegion': null,
    'ShipCountry': 'Norway'
  },
  {
    'OrderID': 10832,
    'CustomerID': 'LAMAI',
    'OrderDate': '1998-01-14T00:00:00.000Z',
    'ShippedDate': '1998-01-19T00:00:00.000Z',
    'Freight': 43.26,
    'ShipName': 'La maison d\'Asie',
    'ShipAddress': '1 rue Alsace-Lorraine',
    'ShipCity': 'Toulouse',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10833,
    'CustomerID': 'OTTIK',
    'OrderDate': '1998-01-15T00:00:00.000Z',
    'ShippedDate': '1998-01-23T00:00:00.000Z',
    'Freight': 71.49,
    'ShipName': 'Ottilies Käseladen',
    'ShipAddress': 'Mehrheimerstr. 369',
    'ShipCity': 'Köln',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10834,
    'CustomerID': 'TRADH',
    'OrderDate': '1998-01-15T00:00:00.000Z',
    'ShippedDate': '1998-01-19T00:00:00.000Z',
    'Freight': 29.78,
    'ShipName': 'Tradição Hipermercados',
    'ShipAddress': 'Av. Inês de Castro, 414',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10835,
    'CustomerID': 'ALFKI',
    'OrderDate': '1998-01-15T00:00:00.000Z',

```

```

    'ShippedDate': '1998-01-21T00:00:00.000Z',
    'Freight': 69.53,
    'ShipName': 'Alfred\' Futterkiste',
    'ShipAddress': 'Obere Str. 57',
    'ShipCity': 'Berlin',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10836,
    'CustomerID': 'ERNSH',
    'OrderDate': '1998-01-16T00:00:00.000Z',
    'ShippedDate': '1998-01-21T00:00:00.000Z',
    'Freight': 411.88,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10837,
    'CustomerID': 'BERGS',
    'OrderDate': '1998-01-16T00:00:00.000Z',
    'ShippedDate': '1998-01-23T00:00:00.000Z',
    'Freight': 13.32,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10838,
    'CustomerID': 'LINOD',
    'OrderDate': '1998-01-19T00:00:00.000Z',
    'ShippedDate': '1998-01-23T00:00:00.000Z',
    'Freight': 59.28,
    'ShipName': 'LINO-Delicateses',
    'ShipAddress': 'Ave. 5 de Mayo Porlamar',
    'ShipCity': 'I. de Margarita',
    'ShipRegion': 'Nueva Esparta',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10839,
    'CustomerID': 'TRADH',
    'OrderDate': '1998-01-19T00:00:00.000Z',
    'ShippedDate': '1998-01-22T00:00:00.000Z',
    'Freight': 35.43,
    'ShipName': 'Tradiçao Hipermercados',
    'ShipAddress': 'Av. Inês de Castro, 414',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {

```

```

    'OrderID': 10840,
    'CustomerID': 'LINOD',
    'OrderDate': '1998-01-19T00:00:00.000Z',
    'ShippedDate': '1998-02-16T00:00:00.000Z',
    'Freight': 2.71,
    'ShipName': 'LINO-Delicateses',
    'ShipAddress': 'Ave. 5 de Mayo Porlamar',
    'ShipCity': 'I. de Margarita',
    'ShipRegion': 'Nueva Esparta',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10841,
    'CustomerID': 'SUPRD',
    'OrderDate': '1998-01-20T00:00:00.000Z',
    'ShippedDate': '1998-01-29T00:00:00.000Z',
    'Freight': 424.3,
    'ShipName': 'Suprêmes délices',
    'ShipAddress': 'Boulevard Tirou, 255',
    'ShipCity': 'Charleroi',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10842,
    'CustomerID': 'TORTU',
    'OrderDate': '1998-01-20T00:00:00.000Z',
    'ShippedDate': '1998-01-29T00:00:00.000Z',
    'Freight': 54.42,
    'ShipName': 'Tortuga Restaurante',
    'ShipAddress': 'Avda. Azteca 123',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10843,
    'CustomerID': 'VICTE',
    'OrderDate': '1998-01-21T00:00:00.000Z',
    'ShippedDate': '1998-01-26T00:00:00.000Z',
    'Freight': 9.26,
    'ShipName': 'Victuailles en stock',
    'ShipAddress': '2, rue du Commerce',
    'ShipCity': 'Lyon',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10844,
    'CustomerID': 'PICCO',
    'OrderDate': '1998-01-21T00:00:00.000Z',
    'ShippedDate': '1998-01-26T00:00:00.000Z',
    'Freight': 25.22,
    'ShipName': 'Piccolo und mehr',
    'ShipAddress': 'Geislweg 14',
    'ShipCity': 'Salzburg',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10845,
    'CustomerID': 'QUICK',
    'OrderDate': '1998-01-21T00:00:00.000Z',
    'ShippedDate': '1998-01-30T00:00:00.000Z',
    'Freight': 212.98,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10846,
    'CustomerID': 'SUPRD',
    'OrderDate': '1998-01-22T00:00:00.000Z',
    'ShippedDate': '1998-01-23T00:00:00.000Z',
    'Freight': 56.46,
    'ShipName': 'Suprêmes délices',
    'ShipAddress': 'Boulevard Tirou, 255',
    'ShipCity': 'Charleroi',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10847,
    'CustomerID': 'SAVEA',
    'OrderDate': '1998-01-22T00:00:00.000Z',
    'ShippedDate': '1998-02-10T00:00:00.000Z',
    'Freight': 487.57,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10848,
    'CustomerID': 'CONSH',
    'OrderDate': '1998-01-23T00:00:00.000Z',
    'ShippedDate': '1998-01-29T00:00:00.000Z',
    'Freight': 38.24,
    'ShipName': 'Consolidated Holdings',
    'ShipAddress': 'Berkeley Gardens 12 Brewery',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10849,
    'CustomerID': 'KOENE',
    'OrderDate': '1998-01-23T00:00:00.000Z',
    'ShippedDate': '1998-01-30T00:00:00.000Z',
    'Freight': 0.56,
    'ShipName': 'Königlich Essen',

```

```

    'ShipAddress': 'Maubelstr. 90',
    'ShipCity': 'Brandenburg',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10850,
    'CustomerID': 'VICTE',
    'OrderDate': '1998-01-23T00:00:00.000Z',
    'ShippedDate': '1998-01-30T00:00:00.000Z',
    'Freight': 49.19,
    'ShipName': 'Victuailles en stock',
    'ShipAddress': '2, rue du Commerce',
    'ShipCity': 'Lyon',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10851,
    'CustomerID': 'RICAR',
    'OrderDate': '1998-01-26T00:00:00.000Z',
    'ShippedDate': '1998-02-02T00:00:00.000Z',
    'Freight': 160.55,
    'ShipName': 'Ricardo Adocicados',
    'ShipAddress': 'Av. Copacabana, 267',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10852,
    'CustomerID': 'RATTC',
    'OrderDate': '1998-01-26T00:00:00.000Z',
    'ShippedDate': '1998-01-30T00:00:00.000Z',
    'Freight': 174.05,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10853,
    'CustomerID': 'BLAUS',
    'OrderDate': '1998-01-27T00:00:00.000Z',
    'ShippedDate': '1998-02-03T00:00:00.000Z',
    'Freight': 53.83,
    'ShipName': 'Blauer See Delikatessen',
    'ShipAddress': 'Forsterstr. 57',
    'ShipCity': 'Mannheim',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10854,
    'CustomerID': 'ERNSH',
    'OrderDate': '1998-01-27T00:00:00.000Z',

```

```

        'ShippedDate': '1998-02-05T00:00:00.000Z',
        'Freight': 100.22,
        'ShipName': 'Ernst Handel',
        'ShipAddress': 'Kirchgasse 6',
        'ShipCity': 'Graz',
        'ShipRegion': null,
        'ShipCountry': 'Austria'
    },
    {
        'OrderID': 10855,
        'CustomerID': 'OLDWO',
        'OrderDate': '1998-01-27T00:00:00.000Z',
        'ShippedDate': '1998-02-04T00:00:00.000Z',
        'Freight': 170.97,
        'ShipName': 'Old World Delicatessen',
        'ShipAddress': '2743 Bering St.',
        'ShipCity': 'Anchorage',
        'ShipRegion': 'AK',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10856,
        'CustomerID': 'ANTON',
        'OrderDate': '1998-01-28T00:00:00.000Z',
        'ShippedDate': '1998-02-10T00:00:00.000Z',
        'Freight': 58.43,
        'ShipName': 'Antonio Moreno Taquería',
        'ShipAddress': 'Mataderos 2312',
        'ShipCity': 'México D.F.',
        'ShipRegion': null,
        'ShipCountry': 'Mexico'
    },
    {
        'OrderID': 10857,
        'CustomerID': 'BERGS',
        'OrderDate': '1998-01-28T00:00:00.000Z',
        'ShippedDate': '1998-02-06T00:00:00.000Z',
        'Freight': 188.85,
        'ShipName': 'Berglunds snabbköp',
        'ShipAddress': 'Berguvsvägen 8',
        'ShipCity': 'Luleå',
        'ShipRegion': null,
        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 10858,
        'CustomerID': 'LACOR',
        'OrderDate': '1998-01-29T00:00:00.000Z',
        'ShippedDate': '1998-02-03T00:00:00.000Z',
        'Freight': 52.51,
        'ShipName': 'La corne d\'abondance',
        'ShipAddress': '67, avenue de l\'Europe',
        'ShipCity': 'Versailles',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {

```

```

    'OrderID': 10859,
    'CustomerID': 'FRANK',
    'OrderDate': '1998-01-29T00:00:00.000Z',
    'ShippedDate': '1998-02-02T00:00:00.000Z',
    'Freight': 76.1,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10860,
    'CustomerID': 'FRANR',
    'OrderDate': '1998-01-29T00:00:00.000Z',
    'ShippedDate': '1998-02-04T00:00:00.000Z',
    'Freight': 19.26,
    'ShipName': 'France restauration',
    'ShipAddress': '54, rue Royale',
    'ShipCity': 'Nantes',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10861,
    'CustomerID': 'WHITC',
    'OrderDate': '1998-01-30T00:00:00.000Z',
    'ShippedDate': '1998-02-17T00:00:00.000Z',
    'Freight': 14.93,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10862,
    'CustomerID': 'LEHMS',
    'OrderDate': '1998-01-30T00:00:00.000Z',
    'ShippedDate': '1998-02-02T00:00:00.000Z',
    'Freight': 53.23,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10863,
    'CustomerID': 'HILAA',
    'OrderDate': '1998-02-02T00:00:00.000Z',
    'ShippedDate': '1998-02-17T00:00:00.000Z',
    'Freight': 30.26,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',

```

```

    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10864,
    'CustomerID': 'AROUT',
    'OrderDate': '1998-02-02T00:00:00.000Z',
    'ShippedDate': '1998-02-09T00:00:00.000Z',
    'Freight': 3.04,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10865,
    'CustomerID': 'QUICK',
    'OrderDate': '1998-02-02T00:00:00.000Z',
    'ShippedDate': '1998-02-12T00:00:00.000Z',
    'Freight': 348.14,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10866,
    'CustomerID': 'BERGS',
    'OrderDate': '1998-02-03T00:00:00.000Z',
    'ShippedDate': '1998-02-12T00:00:00.000Z',
    'Freight': 109.11,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10867,
    'CustomerID': 'LONEP',
    'OrderDate': '1998-02-03T00:00:00.000Z',
    'ShippedDate': '1998-02-11T00:00:00.000Z',
    'Freight': 1.93,
    'ShipName': 'Lonesome Pine Restaurant',
    'ShipAddress': '89 Chiaroscuro Rd.',
    'ShipCity': 'Portland',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10868,
    'CustomerID': 'QUEEN',
    'OrderDate': '1998-02-04T00:00:00.000Z',
    'ShippedDate': '1998-02-23T00:00:00.000Z',
    'Freight': 191.27,
    'ShipName': 'Queen Cozinha',

```



```

      'ShipAddress': 'Alameda dos Canários, 891',
      'ShipCity': 'Sao Paulo',
      'ShipRegion': 'SP',
      'ShipCountry': 'Brazil'
    },
    {
      'OrderID': 10869,
      'CustomerID': 'SEVES',
      'OrderDate': '1998-02-04T00:00:00.000Z',
      'ShippedDate': '1998-02-09T00:00:00.000Z',
      'Freight': 143.28,
      'ShipName': 'Seven Seas Imports',
      'ShipAddress': '90 Wadhurst Rd.',
      'ShipCity': 'London',
      'ShipRegion': null,
      'ShipCountry': 'UK'
    },
    {
      'OrderID': 10870,
      'CustomerID': 'WOLZA',
      'OrderDate': '1998-02-04T00:00:00.000Z',
      'ShippedDate': '1998-02-13T00:00:00.000Z',
      'Freight': 12.04,
      'ShipName': 'Wolski Zajazd',
      'ShipAddress': 'ul. Filtrowa 68',
      'ShipCity': 'Warszawa',
      'ShipRegion': null,
      'ShipCountry': 'Poland'
    },
    {
      'OrderID': 10871,
      'CustomerID': 'BONAP',
      'OrderDate': '1998-02-05T00:00:00.000Z',
      'ShippedDate': '1998-02-10T00:00:00.000Z',
      'Freight': 112.27,
      'ShipName': 'Bon app',
      'ShipAddress': '12, rue des Bouchers',
      'ShipCity': 'Marseille',
      'ShipRegion': null,
      'ShipCountry': 'France'
    },
    {
      'OrderID': 10872,
      'CustomerID': 'GODOS',
      'OrderDate': '1998-02-05T00:00:00.000Z',
      'ShippedDate': '1998-02-09T00:00:00.000Z',
      'Freight': 175.32,
      'ShipName': 'Godos Cocina Típica',
      'ShipAddress': 'C/ Romero, 33',
      'ShipCity': 'Sevilla',
      'ShipRegion': null,
      'ShipCountry': 'Spain'
    },
    {
      'OrderID': 10873,
      'CustomerID': 'WILMK',
      'OrderDate': '1998-02-06T00:00:00.000Z',

```

```

    'ShippedDate': '1998-02-09T00:00:00.000Z',
    'Freight': 0.82,
    'ShipName': 'Wilman Kala',
    'ShipAddress': 'Keskuskatu 45',
    'ShipCity': 'Helsinki',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10874,
    'CustomerID': 'GODOS',
    'OrderDate': '1998-02-06T00:00:00.000Z',
    'ShippedDate': '1998-02-11T00:00:00.000Z',
    'Freight': 19.58,
    'ShipName': 'Godos Cocina Típica',
    'ShipAddress': 'C/ Romero, 33',
    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10875,
    'CustomerID': 'BERGS',
    'OrderDate': '1998-02-06T00:00:00.000Z',
    'ShippedDate': '1998-03-03T00:00:00.000Z',
    'Freight': 32.37,
    'ShipName': 'Berglunds snabbköp',
    'ShipAddress': 'Berguvsvägen 8',
    'ShipCity': 'Luleå',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10876,
    'CustomerID': 'BONAP',
    'OrderDate': '1998-02-09T00:00:00.000Z',
    'ShippedDate': '1998-02-12T00:00:00.000Z',
    'Freight': 60.42,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10877,
    'CustomerID': 'RICAR',
    'OrderDate': '1998-02-09T00:00:00.000Z',
    'ShippedDate': '1998-02-19T00:00:00.000Z',
    'Freight': 38.06,
    'ShipName': 'Ricardo Adocicados',
    'ShipAddress': 'Av. Copacabana, 267',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {

```

```

    'OrderID': 10878,
    'CustomerID': 'QUICK',
    'OrderDate': '1998-02-10T00:00:00.000Z',
    'ShippedDate': '1998-02-12T00:00:00.000Z',
    'Freight': 46.69,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10879,
    'CustomerID': 'WILMK',
    'OrderDate': '1998-02-10T00:00:00.000Z',
    'ShippedDate': '1998-02-12T00:00:00.000Z',
    'Freight': 8.5,
    'ShipName': 'Wilman Kala',
    'ShipAddress': 'Keskuskatu 45',
    'ShipCity': 'Helsinki',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 10880,
    'CustomerID': 'FOLKO',
    'OrderDate': '1998-02-10T00:00:00.000Z',
    'ShippedDate': '1998-02-18T00:00:00.000Z',
    'Freight': 88.01,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10881,
    'CustomerID': 'CACTU',
    'OrderDate': '1998-02-11T00:00:00.000Z',
    'ShippedDate': '1998-02-18T00:00:00.000Z',
    'Freight': 2.84,
    'ShipName': 'Cactus Comidas para llevar',
    'ShipAddress': 'Cerrito 333',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 10882,
    'CustomerID': 'SAVEA',
    'OrderDate': '1998-02-11T00:00:00.000Z',
    'ShippedDate': '1998-02-20T00:00:00.000Z',
    'Freight': 23.1,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',

```

```

    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10883,
    'CustomerID': 'LONEP',
    'OrderDate': '1998-02-12T00:00:00.000Z',
    'ShippedDate': '1998-02-20T00:00:00.000Z',
    'Freight': 0.53,
    'ShipName': 'Lonesome Pine Restaurant',
    'ShipAddress': '89 Chiaroscuro Rd.',
    'ShipCity': 'Portland',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10884,
    'CustomerID': 'LETSS',
    'OrderDate': '1998-02-12T00:00:00.000Z',
    'ShippedDate': '1998-02-13T00:00:00.000Z',
    'Freight': 90.97,
    'ShipName': 'Let\ Stop N Shop',
    'ShipAddress': '87 Polk St. Suite 5',
    'ShipCity': 'San Francisco',
    'ShipRegion': 'CA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10885,
    'CustomerID': 'SUPRD',
    'OrderDate': '1998-02-12T00:00:00.000Z',
    'ShippedDate': '1998-02-18T00:00:00.000Z',
    'Freight': 5.64,
    'ShipName': 'Suprêmes délices',
    'ShipAddress': 'Boulevard Tirou, 255',
    'ShipCity': 'Charleroi',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10886,
    'CustomerID': 'HANAR',
    'OrderDate': '1998-02-13T00:00:00.000Z',
    'ShippedDate': '1998-03-02T00:00:00.000Z',
    'Freight': 4.99,
    'ShipName': 'Hanari Carnes',
    'ShipAddress': 'Rua do Paço, 67',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10887,
    'CustomerID': 'GALED',
    'OrderDate': '1998-02-13T00:00:00.000Z',
    'ShippedDate': '1998-02-16T00:00:00.000Z',
    'Freight': 1.25,
    'ShipName': 'Galería del gastronómo',

```

```

    'ShipAddress': 'Rambla de Cataluña, 23',
    'ShipCity': 'Barcelona',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10888,
    'CustomerID': 'GODOS',
    'OrderDate': '1998-02-16T00:00:00.000Z',
    'ShippedDate': '1998-02-23T00:00:00.000Z',
    'Freight': 51.87,
    'ShipName': 'Godos Cocina Típica',
    'ShipAddress': 'C/ Romero, 33',
    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10889,
    'CustomerID': 'RATTC',
    'OrderDate': '1998-02-16T00:00:00.000Z',
    'ShippedDate': '1998-02-23T00:00:00.000Z',
    'Freight': 280.61,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10890,
    'CustomerID': 'DUMON',
    'OrderDate': '1998-02-16T00:00:00.000Z',
    'ShippedDate': '1998-02-18T00:00:00.000Z',
    'Freight': 32.76,
    'ShipName': 'Du monde entier',
    'ShipAddress': '67, rue des Cinquante Otages',
    'ShipCity': 'Nantes',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10891,
    'CustomerID': 'LEHMS',
    'OrderDate': '1998-02-17T00:00:00.000Z',
    'ShippedDate': '1998-02-19T00:00:00.000Z',
    'Freight': 20.37,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10892,
    'CustomerID': 'MAISD',
    'OrderDate': '1998-02-17T00:00:00.000Z',

```

```

    'ShippedDate': '1998-02-19T00:00:00.000Z',
    'Freight': 120.27,
    'ShipName': 'Maison Dewey',
    'ShipAddress': 'Rue Joseph-Bens 532',
    'ShipCity': 'Bruxelles',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10893,
    'CustomerID': 'KOENE',
    'OrderDate': '1998-02-18T00:00:00.000Z',
    'ShippedDate': '1998-02-20T00:00:00.000Z',
    'Freight': 77.78,
    'ShipName': 'Königlich Essen',
    'ShipAddress': 'Maubelstr. 90',
    'ShipCity': 'Brandenburg',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10894,
    'CustomerID': 'SAVEA',
    'OrderDate': '1998-02-18T00:00:00.000Z',
    'ShippedDate': '1998-02-20T00:00:00.000Z',
    'Freight': 116.13,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10895,
    'CustomerID': 'ERNSH',
    'OrderDate': '1998-02-18T00:00:00.000Z',
    'ShippedDate': '1998-02-23T00:00:00.000Z',
    'Freight': 162.75,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10896,
    'CustomerID': 'MAISD',
    'OrderDate': '1998-02-19T00:00:00.000Z',
    'ShippedDate': '1998-02-27T00:00:00.000Z',
    'Freight': 32.45,
    'ShipName': 'Maison Dewey',
    'ShipAddress': 'Rue Joseph-Bens 532',
    'ShipCity': 'Bruxelles',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {

```

```

    'OrderID': 10897,
    'CustomerID': 'HUNGO',
    'OrderDate': '1998-02-19T00:00:00.000Z',
    'ShippedDate': '1998-02-25T00:00:00.000Z',
    'Freight': 603.54,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10898,
    'CustomerID': 'OCEAN',
    'OrderDate': '1998-02-20T00:00:00.000Z',
    'ShippedDate': '1998-03-06T00:00:00.000Z',
    'Freight': 1.27,
    'ShipName': 'Océano Atlántico Ltda.',
    'ShipAddress': 'Ing. Gustavo Moncada 8585 Piso 20-A',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 10899,
    'CustomerID': 'LILAS',
    'OrderDate': '1998-02-20T00:00:00.000Z',
    'ShippedDate': '1998-02-26T00:00:00.000Z',
    'Freight': 1.21,
    'ShipName': 'LILA-Supermercado',
    'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10900,
    'CustomerID': 'WELLI',
    'OrderDate': '1998-02-20T00:00:00.000Z',
    'ShippedDate': '1998-03-04T00:00:00.000Z',
    'Freight': 1.66,
    'ShipName': 'Wellington Importadora',
    'ShipAddress': 'Rua do Mercado, 12',
    'ShipCity': 'Resende',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10901,
    'CustomerID': 'HILAA',
    'OrderDate': '1998-02-23T00:00:00.000Z',
    'ShippedDate': '1998-02-26T00:00:00.000Z',
    'Freight': 62.09,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
  }

```

```

    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10902,
    'CustomerID': 'FOLKO',
    'OrderDate': '1998-02-23T00:00:00.000Z',
    'ShippedDate': '1998-03-03T00:00:00.000Z',
    'Freight': 44.15,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10903,
    'CustomerID': 'HANAR',
    'OrderDate': '1998-02-24T00:00:00.000Z',
    'ShippedDate': '1998-03-04T00:00:00.000Z',
    'Freight': 36.71,
    'ShipName': 'Hanari Carnes',
    'ShipAddress': 'Rua do Paço, 67',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10904,
    'CustomerID': 'WHITC',
    'OrderDate': '1998-02-24T00:00:00.000Z',
    'ShippedDate': '1998-02-27T00:00:00.000Z',
    'Freight': 162.95,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10905,
    'CustomerID': 'WELLI',
    'OrderDate': '1998-02-24T00:00:00.000Z',
    'ShippedDate': '1998-03-06T00:00:00.000Z',
    'Freight': 13.72,
    'ShipName': 'Wellington Importadora',
    'ShipAddress': 'Rua do Mercado, 12',
    'ShipCity': 'Resende',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10906,
    'CustomerID': 'WOLZA',
    'OrderDate': '1998-02-25T00:00:00.000Z',
    'ShippedDate': '1998-03-03T00:00:00.000Z',
    'Freight': 26.29,
    'ShipName': 'Wolski Zajazd',

```



```

        'ShipAddress': 'ul. Filtrowa 68',
        'ShipCity': 'Warszawa',
        'ShipRegion': null,
        'ShipCountry': 'Poland'
    },
    {
        'OrderID': 10907,
        'CustomerID': 'SPEC'D',
        'OrderDate': '1998-02-25T00:00:00.000Z',
        'ShippedDate': '1998-02-27T00:00:00.000Z',
        'Freight': 9.19,
        'ShipName': 'Spécialités du monde',
        'ShipAddress': '25, rue Lauriston',
        'ShipCity': 'Paris',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10908,
        'CustomerID': 'REGGC',
        'OrderDate': '1998-02-26T00:00:00.000Z',
        'ShippedDate': '1998-03-06T00:00:00.000Z',
        'Freight': 32.96,
        'ShipName': 'Reggiani Caseifici',
        'ShipAddress': 'Strada Provinciale 124',
        'ShipCity': 'Reggio Emilia',
        'ShipRegion': null,
        'ShipCountry': 'Italy'
    },
    {
        'OrderID': 10909,
        'CustomerID': 'SANTG',
        'OrderDate': '1998-02-26T00:00:00.000Z',
        'ShippedDate': '1998-03-10T00:00:00.000Z',
        'Freight': 53.05,
        'ShipName': 'Santé Gourmet',
        'ShipAddress': 'Erling Skakkes gate 78',
        'ShipCity': 'Stavern',
        'ShipRegion': null,
        'ShipCountry': 'Norway'
    },
    {
        'OrderID': 10910,
        'CustomerID': 'WILMK',
        'OrderDate': '1998-02-26T00:00:00.000Z',
        'ShippedDate': '1998-03-04T00:00:00.000Z',
        'Freight': 38.11,
        'ShipName': 'Wilman Kala',
        'ShipAddress': 'Keskuskatu 45',
        'ShipCity': 'Helsinki',
        'ShipRegion': null,
        'ShipCountry': 'Finland'
    },
    {
        'OrderID': 10911,
        'CustomerID': 'GODOS',
        'OrderDate': '1998-02-26T00:00:00.000Z',

```

```

    'ShippedDate': '1998-03-05T00:00:00.000Z',
    'Freight': 38.19,
    'ShipName': 'Godos Cocina Típica',
    'ShipAddress': 'C/ Romero, 33',
    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10912,
    'CustomerID': 'HUNGO',
    'OrderDate': '1998-02-26T00:00:00.000Z',
    'ShippedDate': '1998-03-18T00:00:00.000Z',
    'Freight': 580.91,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10913,
    'CustomerID': 'QUEEN',
    'OrderDate': '1998-02-26T00:00:00.000Z',
    'ShippedDate': '1998-03-04T00:00:00.000Z',
    'Freight': 33.05,
    'ShipName': 'Queen Cozinha',
    'ShipAddress': 'Alameda dos Canários, 891',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10914,
    'CustomerID': 'QUEEN',
    'OrderDate': '1998-02-27T00:00:00.000Z',
    'ShippedDate': '1998-03-02T00:00:00.000Z',
    'Freight': 21.19,
    'ShipName': 'Queen Cozinha',
    'ShipAddress': 'Alameda dos Canários, 891',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10915,
    'CustomerID': 'TORTU',
    'OrderDate': '1998-02-27T00:00:00.000Z',
    'ShippedDate': '1998-03-02T00:00:00.000Z',
    'Freight': 3.51,
    'ShipName': 'Tortuga Restaurante',
    'ShipAddress': 'Avda. Azteca 123',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {

```

```

    'OrderID': 10916,
    'CustomerID': 'RANCH',
    'OrderDate': '1998-02-27T00:00:00.000Z',
    'ShippedDate': '1998-03-09T00:00:00.000Z',
    'Freight': 63.77,
    'ShipName': 'Rancho grande',
    'ShipAddress': 'Av. del Libertador 900',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 10917,
    'CustomerID': 'ROMEY',
    'OrderDate': '1998-03-02T00:00:00.000Z',
    'ShippedDate': '1998-03-11T00:00:00.000Z',
    'Freight': 8.29,
    'ShipName': 'Romero y tomillo',
    'ShipAddress': 'Gran Vía, 1',
    'ShipCity': 'Madrid',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10918,
    'CustomerID': 'BOTTM',
    'OrderDate': '1998-03-02T00:00:00.000Z',
    'ShippedDate': '1998-03-11T00:00:00.000Z',
    'Freight': 48.83,
    'ShipName': 'Bottom-Dollar Markets',
    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10919,
    'CustomerID': 'LINOD',
    'OrderDate': '1998-03-02T00:00:00.000Z',
    'ShippedDate': '1998-03-04T00:00:00.000Z',
    'Freight': 19.8,
    'ShipName': 'LINO-Delicateses',
    'ShipAddress': 'Ave. 5 de Mayo Porlamar',
    'ShipCity': 'I. de Margarita',
    'ShipRegion': 'Nueva Esparta',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10920,
    'CustomerID': 'AROUT',
    'OrderDate': '1998-03-03T00:00:00.000Z',
    'ShippedDate': '1998-03-09T00:00:00.000Z',
    'Freight': 29.61,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',

```

```

        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10921,
        'CustomerID': 'VAFFE',
        'OrderDate': '1998-03-03T00:00:00.000Z',
        'ShippedDate': '1998-03-09T00:00:00.000Z',
        'Freight': 176.48,
        'ShipName': 'Vaffeljernet',
        'ShipAddress': 'Smagsloget 45',
        'ShipCity': 'Århus',
        'ShipRegion': null,
        'ShipCountry': 'Denmark'
    },
    {
        'OrderID': 10922,
        'CustomerID': 'HANAR',
        'OrderDate': '1998-03-03T00:00:00.000Z',
        'ShippedDate': '1998-03-05T00:00:00.000Z',
        'Freight': 62.74,
        'ShipName': 'Hanari Carnes',
        'ShipAddress': 'Rua do Paço, 67',
        'ShipCity': 'Rio de Janeiro',
        'ShipRegion': 'RJ',
        'ShipCountry': 'Brazil'
    },
    {
        'OrderID': 10923,
        'CustomerID': 'LAMAI',
        'OrderDate': '1998-03-03T00:00:00.000Z',
        'ShippedDate': '1998-03-13T00:00:00.000Z',
        'Freight': 68.26,
        'ShipName': 'La maison d\'Asie',
        'ShipAddress': '1 rue Alsace-Lorraine',
        'ShipCity': 'Toulouse',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10924,
        'CustomerID': 'BERGS',
        'OrderDate': '1998-03-04T00:00:00.000Z',
        'ShippedDate': '1998-04-08T00:00:00.000Z',
        'Freight': 151.52,
        'ShipName': 'Berglunds snabbköp',
        'ShipAddress': 'Berguvsvägen 8',
        'ShipCity': 'Luleå',
        'ShipRegion': null,
        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 10925,
        'CustomerID': 'HANAR',
        'OrderDate': '1998-03-04T00:00:00.000Z',
        'ShippedDate': '1998-03-13T00:00:00.000Z',
        'Freight': 2.27,
        'ShipName': 'Hanari Carnes',

```

```

    'ShipAddress': 'Rua do Paço, 67',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10926,
    'CustomerID': 'ANATR',
    'OrderDate': '1998-03-04T00:00:00.000Z',
    'ShippedDate': '1998-03-11T00:00:00.000Z',
    'Freight': 39.92,
    'ShipName': 'Ana Trujillo Emparedados y helados',
    'ShipAddress': 'Avda. de la Constitución 2222',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10927,
    'CustomerID': 'LACOR',
    'OrderDate': '1998-03-05T00:00:00.000Z',
    'ShippedDate': '1998-04-08T00:00:00.000Z',
    'Freight': 19.79,
    'ShipName': 'La corne d\'abondance',
    'ShipAddress': '67, avenue de l\'Europe',
    'ShipCity': 'Versailles',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10928,
    'CustomerID': 'GALED',
    'OrderDate': '1998-03-05T00:00:00.000Z',
    'ShippedDate': '1998-03-18T00:00:00.000Z',
    'Freight': 1.36,
    'ShipName': 'Galería del gastronómo',
    'ShipAddress': 'Rambla de Cataluña, 23',
    'ShipCity': 'Barcelona',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10929,
    'CustomerID': 'FRANK',
    'OrderDate': '1998-03-05T00:00:00.000Z',
    'ShippedDate': '1998-03-12T00:00:00.000Z',
    'Freight': 33.93,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10930,
    'CustomerID': 'SUPRD',
    'OrderDate': '1998-03-06T00:00:00.000Z',

```

```

    'ShippedDate': '1998-03-18T00:00:00.000Z',
    'Freight': 15.55,
    'ShipName': 'Suprêmes délices',
    'ShipAddress': 'Boulevard Tirou, 255',
    'ShipCity': 'Charleroi',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10931,
    'CustomerID': 'RICSU',
    'OrderDate': '1998-03-06T00:00:00.000Z',
    'ShippedDate': '1998-03-19T00:00:00.000Z',
    'Freight': 13.6,
    'ShipName': 'Richter Supermarkt',
    'ShipAddress': 'Starenweg 5',
    'ShipCity': 'Genève',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10932,
    'CustomerID': 'BONAP',
    'OrderDate': '1998-03-06T00:00:00.000Z',
    'ShippedDate': '1998-03-24T00:00:00.000Z',
    'Freight': 134.64,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10933,
    'CustomerID': 'ISLAT',
    'OrderDate': '1998-03-06T00:00:00.000Z',
    'ShippedDate': '1998-03-16T00:00:00.000Z',
    'Freight': 54.15,
    'ShipName': 'Island Trading',
    'ShipAddress': 'Garden House Crowther Way',
    'ShipCity': 'Cowes',
    'ShipRegion': 'Isle of Wight',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10934,
    'CustomerID': 'LEHMS',
    'OrderDate': '1998-03-09T00:00:00.000Z',
    'ShippedDate': '1998-03-12T00:00:00.000Z',
    'Freight': 32.01,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {

```

```

    'OrderID': 10935,
    'CustomerID': 'WELLI',
    'OrderDate': '1998-03-09T00:00:00.000Z',
    'ShippedDate': '1998-03-18T00:00:00.000Z',
    'Freight': 47.59,
    'ShipName': 'Wellington Importadora',
    'ShipAddress': 'Rua do Mercado, 12',
    'ShipCity': 'Resende',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10936,
    'CustomerID': 'GREAL',
    'OrderDate': '1998-03-09T00:00:00.000Z',
    'ShippedDate': '1998-03-18T00:00:00.000Z',
    'Freight': 33.68,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10937,
    'CustomerID': 'CACTU',
    'OrderDate': '1998-03-10T00:00:00.000Z',
    'ShippedDate': '1998-03-13T00:00:00.000Z',
    'Freight': 31.51,
    'ShipName': 'Cactus Comidas para llevar',
    'ShipAddress': 'Cerrito 333',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 10938,
    'CustomerID': 'QUICK',
    'OrderDate': '1998-03-10T00:00:00.000Z',
    'ShippedDate': '1998-03-16T00:00:00.000Z',
    'Freight': 31.89,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10939,
    'CustomerID': 'MAGAA',
    'OrderDate': '1998-03-10T00:00:00.000Z',
    'ShippedDate': '1998-03-13T00:00:00.000Z',
    'Freight': 76.33,
    'ShipName': 'Magazzini Alimentari Riuniti',
    'ShipAddress': 'Via Ludovico il Moro 22',
    'ShipCity': 'Bergamo',
    'ShipRegion': null,

```

```

        'ShipCountry': 'Italy'
    },
    {
        'OrderID': 10940,
        'CustomerID': 'BONAP',
        'OrderDate': '1998-03-11T00:00:00.000Z',
        'ShippedDate': '1998-03-23T00:00:00.000Z',
        'Freight': 19.77,
        'ShipName': 'Bon app',
        'ShipAddress': '12, rue des Bouchers',
        'ShipCity': 'Marseille',
        'ShipRegion': null,
        'ShipCountry': 'France'
    },
    {
        'OrderID': 10941,
        'CustomerID': 'SAVEA',
        'OrderDate': '1998-03-11T00:00:00.000Z',
        'ShippedDate': '1998-03-20T00:00:00.000Z',
        'Freight': 400.81,
        'ShipName': 'Save-a-lot Markets',
        'ShipAddress': '187 Suffolk Ln.',
        'ShipCity': 'Boise',
        'ShipRegion': 'ID',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 10942,
        'CustomerID': 'REGGC',
        'OrderDate': '1998-03-11T00:00:00.000Z',
        'ShippedDate': '1998-03-18T00:00:00.000Z',
        'Freight': 17.95,
        'ShipName': 'Reggiani Caseifici',
        'ShipAddress': 'Strada Provinciale 124',
        'ShipCity': 'Reggio Emilia',
        'ShipRegion': null,
        'ShipCountry': 'Italy'
    },
    {
        'OrderID': 10943,
        'CustomerID': 'BSBEV',
        'OrderDate': '1998-03-11T00:00:00.000Z',
        'ShippedDate': '1998-03-19T00:00:00.000Z',
        'Freight': 2.17,
        'ShipName': 'B\ Beverages',
        'ShipAddress': 'Fauntleroy Circus',
        'ShipCity': 'London',
        'ShipRegion': null,
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 10944,
        'CustomerID': 'BOTTM',
        'OrderDate': '1998-03-12T00:00:00.000Z',
        'ShippedDate': '1998-03-13T00:00:00.000Z',
        'Freight': 52.92,
        'ShipName': 'Bottom-Dollar Markets',

```



```

    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10945,
    'CustomerID': 'MORGK',
    'OrderDate': '1998-03-12T00:00:00.000Z',
    'ShippedDate': '1998-03-18T00:00:00.000Z',
    'Freight': 10.22,
    'ShipName': 'Morgenstern Gesundkost',
    'ShipAddress': 'Heerstr. 22',
    'ShipCity': 'Leipzig',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10946,
    'CustomerID': 'VAFFE',
    'OrderDate': '1998-03-12T00:00:00.000Z',
    'ShippedDate': '1998-03-19T00:00:00.000Z',
    'Freight': 27.2,
    'ShipName': 'Vaffeljernet',
    'ShipAddress': 'Smagsloget 45',
    'ShipCity': 'Århus',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10947,
    'CustomerID': 'BSBEV',
    'OrderDate': '1998-03-13T00:00:00.000Z',
    'ShippedDate': '1998-03-16T00:00:00.000Z',
    'Freight': 3.26,
    'ShipName': 'B\' Beverages',
    'ShipAddress': 'Fauntleroy Circus',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10948,
    'CustomerID': 'GODOS',
    'OrderDate': '1998-03-13T00:00:00.000Z',
    'ShippedDate': '1998-03-19T00:00:00.000Z',
    'Freight': 23.39,
    'ShipName': 'Godos Cocina Típica',
    'ShipAddress': 'C/ Romero, 33',
    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10949,
    'CustomerID': 'BOTTM',
    'OrderDate': '1998-03-13T00:00:00.000Z',

```

```

    'ShippedDate': '1998-03-17T00:00:00.000Z',
    'Freight': 74.44,
    'ShipName': 'Bottom-Dollar Markets',
    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10950,
    'CustomerID': 'MAGAA',
    'OrderDate': '1998-03-16T00:00:00.000Z',
    'ShippedDate': '1998-03-23T00:00:00.000Z',
    'Freight': 2.5,
    'ShipName': 'Magazzini Alimentari Riuniti',
    'ShipAddress': 'Via Ludovico il Moro 22',
    'ShipCity': 'Bergamo',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 10951,
    'CustomerID': 'RICSU',
    'OrderDate': '1998-03-16T00:00:00.000Z',
    'ShippedDate': '1998-04-07T00:00:00.000Z',
    'Freight': 30.85,
    'ShipName': 'Richter Supermarkt',
    'ShipAddress': 'Starenweg 5',
    'ShipCity': 'Genève',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10952,
    'CustomerID': 'ALFKI',
    'OrderDate': '1998-03-16T00:00:00.000Z',
    'ShippedDate': '1998-03-24T00:00:00.000Z',
    'Freight': 40.42,
    'ShipName': 'Alfred\ Futterkiste',
    'ShipAddress': 'Obere Str. 57',
    'ShipCity': 'Berlin',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10953,
    'CustomerID': 'AROUT',
    'OrderDate': '1998-03-16T00:00:00.000Z',
    'ShippedDate': '1998-03-25T00:00:00.000Z',
    'Freight': 23.72,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',
    'ShipCountry': 'UK'
  },
  {

```

```

    'OrderID': 10954,
    'CustomerID': 'LINOD',
    'OrderDate': '1998-03-17T00:00:00.000Z',
    'ShippedDate': '1998-03-20T00:00:00.000Z',
    'Freight': 27.91,
    'ShipName': 'LINO-Delicateses',
    'ShipAddress': 'Ave. 5 de Mayo Porlamar',
    'ShipCity': 'I. de Margarita',
    'ShipRegion': 'Nueva Esparta',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10955,
    'CustomerID': 'FOLKO',
    'OrderDate': '1998-03-17T00:00:00.000Z',
    'ShippedDate': '1998-03-20T00:00:00.000Z',
    'Freight': 3.26,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10956,
    'CustomerID': 'BLAUS',
    'OrderDate': '1998-03-17T00:00:00.000Z',
    'ShippedDate': '1998-03-20T00:00:00.000Z',
    'Freight': 44.65,
    'ShipName': 'Blauer See Delikatessen',
    'ShipAddress': 'Forsterstr. 57',
    'ShipCity': 'Mannheim',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10957,
    'CustomerID': 'HILAA',
    'OrderDate': '1998-03-18T00:00:00.000Z',
    'ShippedDate': '1998-03-27T00:00:00.000Z',
    'Freight': 105.36,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10958,
    'CustomerID': 'OCEAN',
    'OrderDate': '1998-03-18T00:00:00.000Z',
    'ShippedDate': '1998-03-27T00:00:00.000Z',
    'Freight': 49.56,
    'ShipName': 'Océano Atlántico Ltda.',
    'ShipAddress': 'Ing. Gustavo Moncada 8585 Piso 20-A',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 10959,
    'CustomerID': 'GOURL',
    'OrderDate': '1998-03-18T00:00:00.000Z',
    'ShippedDate': '1998-03-23T00:00:00.000Z',
    'Freight': 4.98,
    'ShipName': 'Gourmet Lanchonetes',
    'ShipAddress': 'Av. Brasil, 442',
    'ShipCity': 'Campinas',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10960,
    'CustomerID': 'HILAA',
    'OrderDate': '1998-03-19T00:00:00.000Z',
    'ShippedDate': '1998-04-08T00:00:00.000Z',
    'Freight': 2.08,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10961,
    'CustomerID': 'QUEEN',
    'OrderDate': '1998-03-19T00:00:00.000Z',
    'ShippedDate': '1998-03-30T00:00:00.000Z',
    'Freight': 104.47,
    'ShipName': 'Queen Cozinha',
    'ShipAddress': 'Alameda dos Canários, 891',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10962,
    'CustomerID': 'QUICK',
    'OrderDate': '1998-03-19T00:00:00.000Z',
    'ShippedDate': '1998-03-23T00:00:00.000Z',
    'Freight': 275.79,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10963,
    'CustomerID': 'FURIB',
    'OrderDate': '1998-03-19T00:00:00.000Z',
    'ShippedDate': '1998-03-26T00:00:00.000Z',
    'Freight': 2.7,
    'ShipName': 'Furia Bacalhau e Frutos do Mar',

```

```

    'ShipAddress': 'Jardim das rosas n. 32',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,
    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 10964,
    'CustomerID': 'SPECB',
    'OrderDate': '1998-03-20T00:00:00.000Z',
    'ShippedDate': '1998-03-24T00:00:00.000Z',
    'Freight': 87.38,
    'ShipName': 'Spécialités du monde',
    'ShipAddress': '25, rue Lauriston',
    'ShipCity': 'Paris',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10965,
    'CustomerID': 'OLDWO',
    'OrderDate': '1998-03-20T00:00:00.000Z',
    'ShippedDate': '1998-03-30T00:00:00.000Z',
    'Freight': 144.38,
    'ShipName': 'Old World Delicatessen',
    'ShipAddress': '2743 Bering St.',
    'ShipCity': 'Anchorage',
    'ShipRegion': 'AK',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10966,
    'CustomerID': 'CHOPS',
    'OrderDate': '1998-03-20T00:00:00.000Z',
    'ShippedDate': '1998-04-08T00:00:00.000Z',
    'Freight': 27.19,
    'ShipName': 'Chop-suey Chinese',
    'ShipAddress': 'Hauptstr. 31',
    'ShipCity': 'Bern',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 10967,
    'CustomerID': 'TOMSP',
    'OrderDate': '1998-03-23T00:00:00.000Z',
    'ShippedDate': '1998-04-02T00:00:00.000Z',
    'Freight': 62.22,
    'ShipName': 'Toms Spezialitäten',
    'ShipAddress': 'Luisenstr. 48',
    'ShipCity': 'Münster',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10968,
    'CustomerID': 'ERNSH',
    'OrderDate': '1998-03-23T00:00:00.000Z',

```

```

    'ShippedDate': '1998-04-01T00:00:00.000Z',
    'Freight': 74.6,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10969,
    'CustomerID': 'COMMI',
    'OrderDate': '1998-03-23T00:00:00.000Z',
    'ShippedDate': '1998-03-30T00:00:00.000Z',
    'Freight': 0.21,
    'ShipName': 'Comércio Mineiro',
    'ShipAddress': 'Av. dos Lusíadas, 23',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10970,
    'CustomerID': 'BOLID',
    'OrderDate': '1998-03-24T00:00:00.000Z',
    'ShippedDate': '1998-04-24T00:00:00.000Z',
    'Freight': 16.16,
    'ShipName': 'Bólido Comidas preparadas',
    'ShipAddress': 'C/ Araquil, 67',
    'ShipCity': 'Madrid',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 10971,
    'CustomerID': 'FRANR',
    'OrderDate': '1998-03-24T00:00:00.000Z',
    'ShippedDate': '1998-04-02T00:00:00.000Z',
    'Freight': 121.82,
    'ShipName': 'France restauration',
    'ShipAddress': '54, rue Royale',
    'ShipCity': 'Nantes',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10972,
    'CustomerID': 'LACOR',
    'OrderDate': '1998-03-24T00:00:00.000Z',
    'ShippedDate': '1998-03-26T00:00:00.000Z',
    'Freight': 0.02,
    'ShipName': 'La corne d\'abondance',
    'ShipAddress': '67, avenue de l\'Europe',
    'ShipCity': 'Versailles',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {

```

```

    'OrderID': 10973,
    'CustomerID': 'LACOR',
    'OrderDate': '1998-03-24T00:00:00.000Z',
    'ShippedDate': '1998-03-27T00:00:00.000Z',
    'Freight': 15.17,
    'ShipName': 'La corne d\'abondance',
    'ShipAddress': '67, avenue de l\'Europe',
    'ShipCity': 'Versailles',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 10974,
    'CustomerID': 'SPLIR',
    'OrderDate': '1998-03-25T00:00:00.000Z',
    'ShippedDate': '1998-04-03T00:00:00.000Z',
    'Freight': 12.96,
    'ShipName': 'Split Rail Beer & Ale',
    'ShipAddress': 'P.O. Box 555',
    'ShipCity': 'Lander',
    'ShipRegion': 'WY',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10975,
    'CustomerID': 'BOTTM',
    'OrderDate': '1998-03-25T00:00:00.000Z',
    'ShippedDate': '1998-03-27T00:00:00.000Z',
    'Freight': 32.27,
    'ShipName': 'Bottom-Dollar Markets',
    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10976,
    'CustomerID': 'HILAA',
    'OrderDate': '1998-03-25T00:00:00.000Z',
    'ShippedDate': '1998-04-03T00:00:00.000Z',
    'Freight': 37.97,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10977,
    'CustomerID': 'FOLKO',
    'OrderDate': '1998-03-26T00:00:00.000Z',
    'ShippedDate': '1998-04-10T00:00:00.000Z',
    'Freight': 208.5,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10978,
    'CustomerID': 'MAISD',
    'OrderDate': '1998-03-26T00:00:00.000Z',
    'ShippedDate': '1998-04-23T00:00:00.000Z',
    'Freight': 32.82,
    'ShipName': 'Maison Dewey',
    'ShipAddress': 'Rue Joseph-Bens 532',
    'ShipCity': 'Bruxelles',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 10979,
    'CustomerID': 'ERNSH',
    'OrderDate': '1998-03-26T00:00:00.000Z',
    'ShippedDate': '1998-03-31T00:00:00.000Z',
    'Freight': 353.07,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10980,
    'CustomerID': 'FOLKO',
    'OrderDate': '1998-03-27T00:00:00.000Z',
    'ShippedDate': '1998-04-17T00:00:00.000Z',
    'Freight': 1.26,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10981,
    'CustomerID': 'HANAR',
    'OrderDate': '1998-03-27T00:00:00.000Z',
    'ShippedDate': '1998-04-02T00:00:00.000Z',
    'Freight': 193.37,
    'ShipName': 'Hanari Carnes',
    'ShipAddress': 'Rua do Paço, 67',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10982,
    'CustomerID': 'BOTTM',
    'OrderDate': '1998-03-27T00:00:00.000Z',
    'ShippedDate': '1998-04-08T00:00:00.000Z',
    'Freight': 14.01,
    'ShipName': 'Bottom-Dollar Markets',

```



```

    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 10983,
    'CustomerID': 'SAVEA',
    'OrderDate': '1998-03-27T00:00:00.000Z',
    'ShippedDate': '1998-04-06T00:00:00.000Z',
    'Freight': 657.54,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10984,
    'CustomerID': 'SAVEA',
    'OrderDate': '1998-03-30T00:00:00.000Z',
    'ShippedDate': '1998-04-03T00:00:00.000Z',
    'Freight': 211.22,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10985,
    'CustomerID': 'HUNGO',
    'OrderDate': '1998-03-30T00:00:00.000Z',
    'ShippedDate': '1998-04-02T00:00:00.000Z',
    'Freight': 91.51,
    'ShipName': 'Hungry Owl All-Night Grocers',
    'ShipAddress': '8 Johnstown Road',
    'ShipCity': 'Cork',
    'ShipRegion': 'Co. Cork',
    'ShipCountry': 'Ireland'
  },
  {
    'OrderID': 10986,
    'CustomerID': 'OCEAN',
    'OrderDate': '1998-03-30T00:00:00.000Z',
    'ShippedDate': '1998-04-21T00:00:00.000Z',
    'Freight': 217.86,
    'ShipName': 'Océano Atlántico Ltda.',
    'ShipAddress': 'Ing. Gustavo Moncada 8585 Piso 20-A',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 10987,
    'CustomerID': 'EASTC',
    'OrderDate': '1998-03-31T00:00:00.000Z',

```

```

    'ShippedDate': '1998-04-06T00:00:00.000Z',
    'Freight': 185.48,
    'ShipName': 'Eastern Connection',
    'ShipAddress': '35 King George',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 10988,
    'CustomerID': 'RATTC',
    'OrderDate': '1998-03-31T00:00:00.000Z',
    'ShippedDate': '1998-04-10T00:00:00.000Z',
    'Freight': 61.14,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10989,
    'CustomerID': 'QUEDE',
    'OrderDate': '1998-03-31T00:00:00.000Z',
    'ShippedDate': '1998-04-02T00:00:00.000Z',
    'Freight': 34.76,
    'ShipName': 'Que Delícia',
    'ShipAddress': 'Rua da Panificadora, 12',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 10990,
    'CustomerID': 'ERNSH',
    'OrderDate': '1998-04-01T00:00:00.000Z',
    'ShippedDate': '1998-04-07T00:00:00.000Z',
    'Freight': 117.61,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 10991,
    'CustomerID': 'QUICK',
    'OrderDate': '1998-04-01T00:00:00.000Z',
    'ShippedDate': '1998-04-07T00:00:00.000Z',
    'Freight': 38.51,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {

```

```

    'OrderID': 10992,
    'CustomerID': 'THEBI',
    'OrderDate': '1998-04-01T00:00:00.000Z',
    'ShippedDate': '1998-04-03T00:00:00.000Z',
    'Freight': 4.27,
    'ShipName': 'The Big Cheese',
    'ShipAddress': '89 Jefferson Way Suite 2',
    'ShipCity': 'Portland',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 10993,
    'CustomerID': 'FOLKO',
    'OrderDate': '1998-04-01T00:00:00.000Z',
    'ShippedDate': '1998-04-10T00:00:00.000Z',
    'Freight': 8.81,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 10994,
    'CustomerID': 'VAFFE',
    'OrderDate': '1998-04-02T00:00:00.000Z',
    'ShippedDate': '1998-04-09T00:00:00.000Z',
    'Freight': 65.53,
    'ShipName': 'Vaffeljernet',
    'ShipAddress': 'Smagsloget 45',
    'ShipCity': 'Århus',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 10995,
    'CustomerID': 'PERIC',
    'OrderDate': '1998-04-02T00:00:00.000Z',
    'ShippedDate': '1998-04-06T00:00:00.000Z',
    'Freight': 46,
    'ShipName': 'Pericles Comidas clásicas',
    'ShipAddress': 'Calle Dr. Jorge Cash 321',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 10996,
    'CustomerID': 'QUICK',
    'OrderDate': '1998-04-02T00:00:00.000Z',
    'ShippedDate': '1998-04-10T00:00:00.000Z',
    'Freight': 1.12,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 10997,
    'CustomerID': 'LILAS',
    'OrderDate': '1998-04-03T00:00:00.000Z',
    'ShippedDate': '1998-04-13T00:00:00.000Z',
    'Freight': 73.91,
    'ShipName': 'LILA-Supermercado',
    'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 10998,
    'CustomerID': 'WOLZA',
    'OrderDate': '1998-04-03T00:00:00.000Z',
    'ShippedDate': '1998-04-17T00:00:00.000Z',
    'Freight': 20.31,
    'ShipName': 'Wolski Zajazd',
    'ShipAddress': 'ul. Filtrowa 68',
    'ShipCity': 'Warszawa',
    'ShipRegion': null,
    'ShipCountry': 'Poland'
  },
  {
    'OrderID': 10999,
    'CustomerID': 'OTTIK',
    'OrderDate': '1998-04-03T00:00:00.000Z',
    'ShippedDate': '1998-04-10T00:00:00.000Z',
    'Freight': 96.35,
    'ShipName': 'Ottilies Käseladen',
    'ShipAddress': 'Mehrheimerstr. 369',
    'ShipCity': 'Köln',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 11000,
    'CustomerID': 'RATTC',
    'OrderDate': '1998-04-06T00:00:00.000Z',
    'ShippedDate': '1998-04-14T00:00:00.000Z',
    'Freight': 55.12,
    'ShipName': 'Rattlesnake Canyon Grocery',
    'ShipAddress': '2817 Milton Dr.',
    'ShipCity': 'Albuquerque',
    'ShipRegion': 'NM',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 11001,
    'CustomerID': 'FOLKO',
    'OrderDate': '1998-04-06T00:00:00.000Z',
    'ShippedDate': '1998-04-14T00:00:00.000Z',
    'Freight': 197.3,
    'ShipName': 'Folk och fä HB',

```

```

        'ShipAddress': 'Åkergatan 24',
        'ShipCity': 'Bräcke',
        'ShipRegion': null,
        'ShipCountry': 'Sweden'
    },
    {
        'OrderID': 11002,
        'CustomerID': 'SAVEA',
        'OrderDate': '1998-04-06T00:00:00.000Z',
        'ShippedDate': '1998-04-16T00:00:00.000Z',
        'Freight': 141.16,
        'ShipName': 'Save-a-lot Markets',
        'ShipAddress': '187 Suffolk Ln.',
        'ShipCity': 'Boise',
        'ShipRegion': 'ID',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 11003,
        'CustomerID': 'THECR',
        'OrderDate': '1998-04-06T00:00:00.000Z',
        'ShippedDate': '1998-04-08T00:00:00.000Z',
        'Freight': 14.91,
        'ShipName': 'The Cracker Box',
        'ShipAddress': '55 Grizzly Peak Rd.',
        'ShipCity': 'Butte',
        'ShipRegion': 'MT',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 11004,
        'CustomerID': 'MAISD',
        'OrderDate': '1998-04-07T00:00:00.000Z',
        'ShippedDate': '1998-04-20T00:00:00.000Z',
        'Freight': 44.84,
        'ShipName': 'Maison Dewey',
        'ShipAddress': 'Rue Joseph-Bens 532',
        'ShipCity': 'Bruxelles',
        'ShipRegion': null,
        'ShipCountry': 'Belgium'
    },
    {
        'OrderID': 11005,
        'CustomerID': 'WILMK',
        'OrderDate': '1998-04-07T00:00:00.000Z',
        'ShippedDate': '1998-04-10T00:00:00.000Z',
        'Freight': 0.75,
        'ShipName': 'Wilman Kala',
        'ShipAddress': 'Keskuskatu 45',
        'ShipCity': 'Helsinki',
        'ShipRegion': null,
        'ShipCountry': 'Finland'
    },
    {
        'OrderID': 11006,
        'CustomerID': 'GREAL',
        'OrderDate': '1998-04-07T00:00:00.000Z',

```

```

    'ShippedDate': '1998-04-15T00:00:00.000Z',
    'Freight': 25.19,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 11007,
    'CustomerID': 'PRINI',
    'OrderDate': '1998-04-08T00:00:00.000Z',
    'ShippedDate': '1998-04-13T00:00:00.000Z',
    'Freight': 202.24,
    'ShipName': 'Princesa Isabel Vinhos',
    'ShipAddress': 'Estrada da saúde n. 58',
    'ShipCity': 'Lisboa',
    'ShipRegion': null,
    'ShipCountry': 'Portugal'
  },
  {
    'OrderID': 11008,
    'CustomerID': 'ERNSH',
    'OrderDate': '1998-04-08T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 79.46,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 11009,
    'CustomerID': 'GODOS',
    'OrderDate': '1998-04-08T00:00:00.000Z',
    'ShippedDate': '1998-04-10T00:00:00.000Z',
    'Freight': 59.11,
    'ShipName': 'Godos Cocina Típica',
    'ShipAddress': 'C/ Romero, 33',
    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 11010,
    'CustomerID': 'REGGC',
    'OrderDate': '1998-04-09T00:00:00.000Z',
    'ShippedDate': '1998-04-21T00:00:00.000Z',
    'Freight': 28.71,
    'ShipName': 'Reggiani Caseifici',
    'ShipAddress': 'Strada Provinciale 124',
    'ShipCity': 'Reggio Emilia',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {

```

```

    'OrderID': 11011,
    'CustomerID': 'ALFKI',
    'OrderDate': '1998-04-09T00:00:00.000Z',
    'ShippedDate': '1998-04-13T00:00:00.000Z',
    'Freight': 1.21,
    'ShipName': 'Alfred\' Futterkiste',
    'ShipAddress': 'Obere Str. 57',
    'ShipCity': 'Berlin',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 11012,
    'CustomerID': 'FRANK',
    'OrderDate': '1998-04-09T00:00:00.000Z',
    'ShippedDate': '1998-04-17T00:00:00.000Z',
    'Freight': 242.95,
    'ShipName': 'Frankenversand',
    'ShipAddress': 'Berliner Platz 43',
    'ShipCity': 'München',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 11013,
    'CustomerID': 'ROMEY',
    'OrderDate': '1998-04-09T00:00:00.000Z',
    'ShippedDate': '1998-04-10T00:00:00.000Z',
    'Freight': 32.99,
    'ShipName': 'Romero y tomillo',
    'ShipAddress': 'Gran Vía, 1',
    'ShipCity': 'Madrid',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 11014,
    'CustomerID': 'LINOD',
    'OrderDate': '1998-04-10T00:00:00.000Z',
    'ShippedDate': '1998-04-15T00:00:00.000Z',
    'Freight': 23.6,
    'ShipName': 'LINO-Delicateses',
    'ShipAddress': 'Ave. 5 de Mayo Porlamar',
    'ShipCity': 'I. de Margarita',
    'ShipRegion': 'Nueva Esparta',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 11015,
    'CustomerID': 'SANTG',
    'OrderDate': '1998-04-10T00:00:00.000Z',
    'ShippedDate': '1998-04-20T00:00:00.000Z',
    'Freight': 4.62,
    'ShipName': 'Santé Gourmet',
    'ShipAddress': 'Erling Skakkes gate 78',
    'ShipCity': 'Stavern',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Norway'
  },
  {
    'OrderID': 11016,
    'CustomerID': 'AROUT',
    'OrderDate': '1998-04-10T00:00:00.000Z',
    'ShippedDate': '1998-04-13T00:00:00.000Z',
    'Freight': 33.8,
    'ShipName': 'Around the Horn',
    'ShipAddress': 'Brook Farm Stratford St. Mary',
    'ShipCity': 'Colchester',
    'ShipRegion': 'Essex',
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 11017,
    'CustomerID': 'ERNSH',
    'OrderDate': '1998-04-13T00:00:00.000Z',
    'ShippedDate': '1998-04-20T00:00:00.000Z',
    'Freight': 754.26,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,
    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 11018,
    'CustomerID': 'LONEP',
    'OrderDate': '1998-04-13T00:00:00.000Z',
    'ShippedDate': '1998-04-16T00:00:00.000Z',
    'Freight': 11.65,
    'ShipName': 'Lonesome Pine Restaurant',
    'ShipAddress': '89 Chiaroscuro Rd.',
    'ShipCity': 'Portland',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 11019,
    'CustomerID': 'RANCH',
    'OrderDate': '1998-04-13T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 3.17,
    'ShipName': 'Rancho grande',
    'ShipAddress': 'Av. del Libertador 900',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 11020,
    'CustomerID': 'OTTIK',
    'OrderDate': '1998-04-14T00:00:00.000Z',
    'ShippedDate': '1998-04-16T00:00:00.000Z',
    'Freight': 43.3,
    'ShipName': 'Ottilies Käseladen',

```



```

    'ShipAddress': 'Mehrheimerstr. 369',
    'ShipCity': 'Köln',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 11021,
    'CustomerID': 'QUICK',
    'OrderDate': '1998-04-14T00:00:00.000Z',
    'ShippedDate': '1998-04-21T00:00:00.000Z',
    'Freight': 297.18,
    'ShipName': 'QUICK-Stop',
    'ShipAddress': 'Taucherstraße 10',
    'ShipCity': 'Cunewalde',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 11022,
    'CustomerID': 'HANAR',
    'OrderDate': '1998-04-14T00:00:00.000Z',
    'ShippedDate': '1998-05-04T00:00:00.000Z',
    'Freight': 6.27,
    'ShipName': 'Hanari Carnes',
    'ShipAddress': 'Rua do Paço, 67',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 11023,
    'CustomerID': 'BSBEV',
    'OrderDate': '1998-04-14T00:00:00.000Z',
    'ShippedDate': '1998-04-24T00:00:00.000Z',
    'Freight': 123.83,
    'ShipName': 'B\' Beverages',
    'ShipAddress': 'Fauntleroy Circus',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 11024,
    'CustomerID': 'EASTC',
    'OrderDate': '1998-04-15T00:00:00.000Z',
    'ShippedDate': '1998-04-20T00:00:00.000Z',
    'Freight': 74.36,
    'ShipName': 'Eastern Connection',
    'ShipAddress': '35 King George',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 11025,
    'CustomerID': 'WARTH',
    'OrderDate': '1998-04-15T00:00:00.000Z',

```

```

    'ShippedDate': '1998-04-24T00:00:00.000Z',
    'Freight': 29.17,
    'ShipName': 'Wartian Herkku',
    'ShipAddress': 'Torikatu 38',
    'ShipCity': 'Oulu',
    'ShipRegion': null,
    'ShipCountry': 'Finland'
  },
  {
    'OrderID': 11026,
    'CustomerID': 'FRANS',
    'OrderDate': '1998-04-15T00:00:00.000Z',
    'ShippedDate': '1998-04-28T00:00:00.000Z',
    'Freight': 47.09,
    'ShipName': 'Franchi S.p.A.',
    'ShipAddress': 'Via Monte Bianco 34',
    'ShipCity': 'Torino',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 11027,
    'CustomerID': 'BOTTM',
    'OrderDate': '1998-04-16T00:00:00.000Z',
    'ShippedDate': '1998-04-20T00:00:00.000Z',
    'Freight': 52.52,
    'ShipName': 'Bottom-Dollar Markets',
    'ShipAddress': '23 Tsawassen Blvd.',
    'ShipCity': 'Tsawassen',
    'ShipRegion': 'BC',
    'ShipCountry': 'Canada'
  },
  {
    'OrderID': 11028,
    'CustomerID': 'KOENE',
    'OrderDate': '1998-04-16T00:00:00.000Z',
    'ShippedDate': '1998-04-22T00:00:00.000Z',
    'Freight': 29.59,
    'ShipName': 'Königlich Essen',
    'ShipAddress': 'Maubelstr. 90',
    'ShipCity': 'Brandenburg',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 11029,
    'CustomerID': 'CHOPS',
    'OrderDate': '1998-04-16T00:00:00.000Z',
    'ShippedDate': '1998-04-27T00:00:00.000Z',
    'Freight': 47.84,
    'ShipName': 'Chop-suey Chinese',
    'ShipAddress': 'Hauptstr. 31',
    'ShipCity': 'Bern',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  }
]

```

```

    'OrderID': 11030,
    'CustomerID': 'SAVEA',
    'OrderDate': '1998-04-17T00:00:00.000Z',
    'ShippedDate': '1998-04-27T00:00:00.000Z',
    'Freight': 830.75,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 11031,
    'CustomerID': 'SAVEA',
    'OrderDate': '1998-04-17T00:00:00.000Z',
    'ShippedDate': '1998-04-24T00:00:00.000Z',
    'Freight': 227.22,
    'ShipName': 'Save-a-lot Markets',
    'ShipAddress': '187 Suffolk Ln.',
    'ShipCity': 'Boise',
    'ShipRegion': 'ID',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 11032,
    'CustomerID': 'WHITC',
    'OrderDate': '1998-04-17T00:00:00.000Z',
    'ShippedDate': '1998-04-23T00:00:00.000Z',
    'Freight': 606.19,
    'ShipName': 'White Clover Markets',
    'ShipAddress': '1029 - 12th Ave. S.',
    'ShipCity': 'Seattle',
    'ShipRegion': 'WA',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 11033,
    'CustomerID': 'RICSU',
    'OrderDate': '1998-04-17T00:00:00.000Z',
    'ShippedDate': '1998-04-23T00:00:00.000Z',
    'Freight': 84.74,
    'ShipName': 'Richter Supermarkt',
    'ShipAddress': 'Starenweg 5',
    'ShipCity': 'Genève',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 11034,
    'CustomerID': 'OLDWO',
    'OrderDate': '1998-04-20T00:00:00.000Z',
    'ShippedDate': '1998-04-27T00:00:00.000Z',
    'Freight': 40.32,
    'ShipName': 'Old World Delicatessen',
    'ShipAddress': '2743 Bering St.',
    'ShipCity': 'Anchorage',
    'ShipRegion': 'AK',
  }

```

```

    'ShipCountry': 'USA'
  },
  {
    'OrderID': 11035,
    'CustomerID': 'SUPRD',
    'OrderDate': '1998-04-20T00:00:00.000Z',
    'ShippedDate': '1998-04-24T00:00:00.000Z',
    'Freight': 0.17,
    'ShipName': 'Suprêmes délices',
    'ShipAddress': 'Boulevard Tirou, 255',
    'ShipCity': 'Charleroi',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 11036,
    'CustomerID': 'DRACD',
    'OrderDate': '1998-04-20T00:00:00.000Z',
    'ShippedDate': '1998-04-22T00:00:00.000Z',
    'Freight': 149.47,
    'ShipName': 'Drachenblut Delikatessen',
    'ShipAddress': 'Walserweg 21',
    'ShipCity': 'Aachen',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 11037,
    'CustomerID': 'GODOS',
    'OrderDate': '1998-04-21T00:00:00.000Z',
    'ShippedDate': '1998-04-27T00:00:00.000Z',
    'Freight': 3.2,
    'ShipName': 'Godos Cocina Típica',
    'ShipAddress': 'C/ Romero, 33',
    'ShipCity': 'Sevilla',
    'ShipRegion': null,
    'ShipCountry': 'Spain'
  },
  {
    'OrderID': 11038,
    'CustomerID': 'SUPRD',
    'OrderDate': '1998-04-21T00:00:00.000Z',
    'ShippedDate': '1998-04-30T00:00:00.000Z',
    'Freight': 29.59,
    'ShipName': 'Suprêmes délices',
    'ShipAddress': 'Boulevard Tirou, 255',
    'ShipCity': 'Charleroi',
    'ShipRegion': null,
    'ShipCountry': 'Belgium'
  },
  {
    'OrderID': 11039,
    'CustomerID': 'LINOD',
    'OrderDate': '1998-04-21T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 65,
    'ShipName': 'LINO-Delicateses',

```

```

    'ShipAddress': 'Ave. 5 de Mayo Porlamar',
    'ShipCity': 'I. de Margarita',
    'ShipRegion': 'Nueva Esparta',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 11040,
    'CustomerID': 'GREAL',
    'OrderDate': '1998-04-22T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 18.84,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 11041,
    'CustomerID': 'CHOPS',
    'OrderDate': '1998-04-22T00:00:00.000Z',
    'ShippedDate': '1998-04-28T00:00:00.000Z',
    'Freight': 48.22,
    'ShipName': 'Chop-suey Chinese',
    'ShipAddress': 'Hauptstr. 31',
    'ShipCity': 'Bern',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 11042,
    'CustomerID': 'COMMI',
    'OrderDate': '1998-04-22T00:00:00.000Z',
    'ShippedDate': '1998-05-01T00:00:00.000Z',
    'Freight': 29.99,
    'ShipName': 'Comércio Mineiro',
    'ShipAddress': 'Av. dos Lusíadas, 23',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 11043,
    'CustomerID': 'SPECB',
    'OrderDate': '1998-04-22T00:00:00.000Z',
    'ShippedDate': '1998-04-29T00:00:00.000Z',
    'Freight': 8.8,
    'ShipName': 'Spécialités du monde',
    'ShipAddress': '25, rue Lauriston',
    'ShipCity': 'Paris',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 11044,
    'CustomerID': 'WOLZA',
    'OrderDate': '1998-04-23T00:00:00.000Z',

```

```

        'ShippedDate': '1998-05-01T00:00:00.000Z',
        'Freight': 8.72,
        'ShipName': 'Wolski Zajazd',
        'ShipAddress': 'ul. Filtrowa 68',
        'ShipCity': 'Warszawa',
        'ShipRegion': null,
        'ShipCountry': 'Poland'
    },
    {
        'OrderID': 11045,
        'CustomerID': 'BOTTM',
        'OrderDate': '1998-04-23T00:00:00.000Z',
        'ShippedDate': null,
        'Freight': 70.58,
        'ShipName': 'Bottom-Dollar Markets',
        'ShipAddress': '23 Tsawassen Blvd.',
        'ShipCity': 'Tsawassen',
        'ShipRegion': 'BC',
        'ShipCountry': 'Canada'
    },
    {
        'OrderID': 11046,
        'CustomerID': 'WANDK',
        'OrderDate': '1998-04-23T00:00:00.000Z',
        'ShippedDate': '1998-04-24T00:00:00.000Z',
        'Freight': 71.64,
        'ShipName': 'Die Wandernde Kuh',
        'ShipAddress': 'Adenauerallee 900',
        'ShipCity': 'Stuttgart',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {
        'OrderID': 11047,
        'CustomerID': 'EASTC',
        'OrderDate': '1998-04-24T00:00:00.000Z',
        'ShippedDate': '1998-05-01T00:00:00.000Z',
        'Freight': 46.62,
        'ShipName': 'Eastern Connection',
        'ShipAddress': '35 King George',
        'ShipCity': 'London',
        'ShipRegion': null,
        'ShipCountry': 'UK'
    },
    {
        'OrderID': 11048,
        'CustomerID': 'BOTTM',
        'OrderDate': '1998-04-24T00:00:00.000Z',
        'ShippedDate': '1998-04-30T00:00:00.000Z',
        'Freight': 24.12,
        'ShipName': 'Bottom-Dollar Markets',
        'ShipAddress': '23 Tsawassen Blvd.',
        'ShipCity': 'Tsawassen',
        'ShipRegion': 'BC',
        'ShipCountry': 'Canada'
    }
]

```

```

    'OrderID': 11049,
    'CustomerID': 'GOURL',
    'OrderDate': '1998-04-24T00:00:00.000Z',
    'ShippedDate': '1998-05-04T00:00:00.000Z',
    'Freight': 8.34,
    'ShipName': 'Gourmet Lanchonetes',
    'ShipAddress': 'Av. Brasil, 442',
    'ShipCity': 'Campinas',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 11050,
    'CustomerID': 'FOLKO',
    'OrderDate': '1998-04-27T00:00:00.000Z',
    'ShippedDate': '1998-05-05T00:00:00.000Z',
    'Freight': 59.41,
    'ShipName': 'Folk och fä HB',
    'ShipAddress': 'Åkergatan 24',
    'ShipCity': 'Bräcke',
    'ShipRegion': null,
    'ShipCountry': 'Sweden'
  },
  {
    'OrderID': 11051,
    'CustomerID': 'LAMAI',
    'OrderDate': '1998-04-27T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 2.79,
    'ShipName': 'La maison d\'Asie',
    'ShipAddress': '1 rue Alsace-Lorraine',
    'ShipCity': 'Toulouse',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 11052,
    'CustomerID': 'HANAR',
    'OrderDate': '1998-04-27T00:00:00.000Z',
    'ShippedDate': '1998-05-01T00:00:00.000Z',
    'Freight': 67.26,
    'ShipName': 'Hanari Carnes',
    'ShipAddress': 'Rua do Paço, 67',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 11053,
    'CustomerID': 'PICCO',
    'OrderDate': '1998-04-27T00:00:00.000Z',
    'ShippedDate': '1998-04-29T00:00:00.000Z',
    'Freight': 53.05,
    'ShipName': 'Piccolo und mehr',
    'ShipAddress': 'Geislweg 14',
    'ShipCity': 'Salzburg',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 11054,
    'CustomerID': 'CACTU',
    'OrderDate': '1998-04-28T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 0.33,
    'ShipName': 'Cactus Comidas para llevar',
    'ShipAddress': 'Cerrito 333',
    'ShipCity': 'Buenos Aires',
    'ShipRegion': null,
    'ShipCountry': 'Argentina'
  },
  {
    'OrderID': 11055,
    'CustomerID': 'HILAA',
    'OrderDate': '1998-04-28T00:00:00.000Z',
    'ShippedDate': '1998-05-05T00:00:00.000Z',
    'Freight': 120.92,
    'ShipName': 'HILARION-Abastos',
    'ShipAddress': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'ShipCity': 'San Cristóbal',
    'ShipRegion': 'Táchira',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 11056,
    'CustomerID': 'EASTC',
    'OrderDate': '1998-04-28T00:00:00.000Z',
    'ShippedDate': '1998-05-01T00:00:00.000Z',
    'Freight': 278.96,
    'ShipName': 'Eastern Connection',
    'ShipAddress': '35 King George',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 11057,
    'CustomerID': 'NORTS',
    'OrderDate': '1998-04-29T00:00:00.000Z',
    'ShippedDate': '1998-05-01T00:00:00.000Z',
    'Freight': 4.13,
    'ShipName': 'North/South',
    'ShipAddress': 'South House 300 Queensbridge',
    'ShipCity': 'London',
    'ShipRegion': null,
    'ShipCountry': 'UK'
  },
  {
    'OrderID': 11058,
    'CustomerID': 'BLAUS',
    'OrderDate': '1998-04-29T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 31.14,
    'ShipName': 'Blauer See Delikatessen',

```



```

    'ShipAddress': 'Forsterstr. 57',
    'ShipCity': 'Mannheim',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 11059,
    'CustomerID': 'RICAR',
    'OrderDate': '1998-04-29T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 85.8,
    'ShipName': 'Ricardo Adocicados',
    'ShipAddress': 'Av. Copacabana, 267',
    'ShipCity': 'Rio de Janeiro',
    'ShipRegion': 'RJ',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 11060,
    'CustomerID': 'FRANS',
    'OrderDate': '1998-04-30T00:00:00.000Z',
    'ShippedDate': '1998-05-04T00:00:00.000Z',
    'Freight': 10.98,
    'ShipName': 'Franchi S.p.A.',
    'ShipAddress': 'Via Monte Bianco 34',
    'ShipCity': 'Torino',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 11061,
    'CustomerID': 'GREAL',
    'OrderDate': '1998-04-30T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 14.01,
    'ShipName': 'Great Lakes Food Market',
    'ShipAddress': '2732 Baker Blvd.',
    'ShipCity': 'Eugene',
    'ShipRegion': 'OR',
    'ShipCountry': 'USA'
  },
  {
    'OrderID': 11062,
    'CustomerID': 'REGGC',
    'OrderDate': '1998-04-30T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 29.93,
    'ShipName': 'Reggiani Caseifici',
    'ShipAddress': 'Strada Provinciale 124',
    'ShipCity': 'Reggio Emilia',
    'ShipRegion': null,
    'ShipCountry': 'Italy'
  },
  {
    'OrderID': 11063,
    'CustomerID': 'HUNGO',
    'OrderDate': '1998-04-30T00:00:00.000Z',

```

```

        'ShippedDate': '1998-05-06T00:00:00.000Z',
        'Freight': 81.73,
        'ShipName': 'Hungry Owl All-Night Grocers',
        'ShipAddress': '8 Johnstown Road',
        'ShipCity': 'Cork',
        'ShipRegion': 'Co. Cork',
        'ShipCountry': 'Ireland'
    },
    {
        'OrderID': 11064,
        'CustomerID': 'SAVEA',
        'OrderDate': '1998-05-01T00:00:00.000Z',
        'ShippedDate': '1998-05-04T00:00:00.000Z',
        'Freight': 30.09,
        'ShipName': 'Save-a-lot Markets',
        'ShipAddress': '187 Suffolk Ln.',
        'ShipCity': 'Boise',
        'ShipRegion': 'ID',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 11065,
        'CustomerID': 'LILAS',
        'OrderDate': '1998-05-01T00:00:00.000Z',
        'ShippedDate': null,
        'Freight': 12.91,
        'ShipName': 'LILA-Supermercado',
        'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
        'ShipCity': 'Barquisimeto',
        'ShipRegion': 'Lara',
        'ShipCountry': 'Venezuela'
    },
    {
        'OrderID': 11066,
        'CustomerID': 'WHITC',
        'OrderDate': '1998-05-01T00:00:00.000Z',
        'ShippedDate': '1998-05-04T00:00:00.000Z',
        'Freight': 44.72,
        'ShipName': 'White Clover Markets',
        'ShipAddress': '1029 - 12th Ave. S.',
        'ShipCity': 'Seattle',
        'ShipRegion': 'WA',
        'ShipCountry': 'USA'
    },
    {
        'OrderID': 11067,
        'CustomerID': 'DRACD',
        'OrderDate': '1998-05-04T00:00:00.000Z',
        'ShippedDate': '1998-05-06T00:00:00.000Z',
        'Freight': 7.98,
        'ShipName': 'Drachenblut Delikatessen',
        'ShipAddress': 'Walserweg 21',
        'ShipCity': 'Aachen',
        'ShipRegion': null,
        'ShipCountry': 'Germany'
    },
    {

```

```

    'OrderID': 11068,
    'CustomerID': 'QUEEN',
    'OrderDate': '1998-05-04T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 81.75,
    'ShipName': 'Queen Cozinha',
    'ShipAddress': 'Alameda dos Canários, 891',
    'ShipCity': 'Sao Paulo',
    'ShipRegion': 'SP',
    'ShipCountry': 'Brazil'
  },
  {
    'OrderID': 11069,
    'CustomerID': 'TORTU',
    'OrderDate': '1998-05-04T00:00:00.000Z',
    'ShippedDate': '1998-05-06T00:00:00.000Z',
    'Freight': 15.67,
    'ShipName': 'Tortuga Restaurante',
    'ShipAddress': 'Avda. Azteca 123',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 11070,
    'CustomerID': 'LEHMS',
    'OrderDate': '1998-05-05T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 136,
    'ShipName': 'Lehmanns Marktstand',
    'ShipAddress': 'Magazinweg 7',
    'ShipCity': 'Frankfurt a.M.',
    'ShipRegion': null,
    'ShipCountry': 'Germany'
  },
  {
    'OrderID': 11071,
    'CustomerID': 'LILAS',
    'OrderDate': '1998-05-05T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 0.93,
    'ShipName': 'LILA-Supermercado',
    'ShipAddress': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'ShipCity': 'Barquisimeto',
    'ShipRegion': 'Lara',
    'ShipCountry': 'Venezuela'
  },
  {
    'OrderID': 11072,
    'CustomerID': 'ERNSH',
    'OrderDate': '1998-05-05T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 258.64,
    'ShipName': 'Ernst Handel',
    'ShipAddress': 'Kirchgasse 6',
    'ShipCity': 'Graz',
    'ShipRegion': null,

```

```

    'ShipCountry': 'Austria'
  },
  {
    'OrderID': 11073,
    'CustomerID': 'PERIC',
    'OrderDate': '1998-05-05T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 24.95,
    'ShipName': 'Pericles Comidas clásicas',
    'ShipAddress': 'Calle Dr. Jorge Cash 321',
    'ShipCity': 'México D.F.',
    'ShipRegion': null,
    'ShipCountry': 'Mexico'
  },
  {
    'OrderID': 11074,
    'CustomerID': 'SIMOB',
    'OrderDate': '1998-05-06T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 18.44,
    'ShipName': 'Simons bistro',
    'ShipAddress': 'Vinbæltet 34',
    'ShipCity': 'Kobenhavn',
    'ShipRegion': null,
    'ShipCountry': 'Denmark'
  },
  {
    'OrderID': 11075,
    'CustomerID': 'RICSU',
    'OrderDate': '1998-05-06T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 6.19,
    'ShipName': 'Richter Supermarkt',
    'ShipAddress': 'Starenweg 5',
    'ShipCity': 'Genève',
    'ShipRegion': null,
    'ShipCountry': 'Switzerland'
  },
  {
    'OrderID': 11076,
    'CustomerID': 'BONAP',
    'OrderDate': '1998-05-06T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 38.28,
    'ShipName': 'Bon app',
    'ShipAddress': '12, rue des Bouchers',
    'ShipCity': 'Marseille',
    'ShipRegion': null,
    'ShipCountry': 'France'
  },
  {
    'OrderID': 11077,
    'CustomerID': 'RATTC',
    'OrderDate': '1998-05-06T00:00:00.000Z',
    'ShippedDate': null,
    'Freight': 8.53,
    'ShipName': 'Rattlesnake Canyon Grocery',

```

```

        'ShipAddress': '2817 Milton Dr.',
        'ShipCity': 'Albuquerque',
        'ShipRegion': 'NM',
        'ShipCountry': 'USA'
    }
});
export const orderData: Object[] = JSON.parse(stringData, (field: any,
value: any) => {
    let dupValue: any = value;
    if (typeof value === 'string' && /^(\\d{4}\\-\\d\\d\\-\\d\\d([tT][\\d:\\.]*){1}) ([zZ]|([+\\-]) (\\d\\d):?(\\d\\d))?$/).test(value)) {
        let arr: any = dupValue.split(/^[^0-9]/);
        let arg: any = parseInt(arr[4], 10);
        let arg1: any = parseInt(arr[5], 10);
        value = new Date(parseInt(arr[0], 10), parseInt(arr[1], 10) - 1,
        parseInt(arr[2], 10), parseInt(arr[3], 10), arg, arg1);
    }
    return value;
});
export const categoryData: Object[] = [
    {
        'CategoryName': 'Beverages',
        'ProductName': 'Chai',
        'QuantityPerUnit': '10 boxes x 20 bags',
        'UnitsInStock': 39,
        'Discontinued': true
    },
    {
        'CategoryName': 'Beverages',
        'ProductName': 'Chang',
        'QuantityPerUnit': '24 - 12 oz bottles',
        'UnitsInStock': 17,
        'Discontinued': true
    },
    {
        'CategoryName': 'Beverages',
        'ProductName': 'Chartreuse verte',
        'QuantityPerUnit': '750 cc per bottle',
        'UnitsInStock': 69,
        'Discontinued': true
    },
    {
        'CategoryName': 'Beverages',
        'ProductName': 'C\\u00f4te de Blaye',
        'QuantityPerUnit': '12 - 75 cl bottles',
        'UnitsInStock': 17,
        'Discontinued': false
    },
    {
        'CategoryName': 'Beverages',
        'ProductName': 'Ipoh Coffee',
        'QuantityPerUnit': '16 - 500 g tins',
        'UnitsInStock': 17,
        'Discontinued': true
    },
    {
        'CategoryName': 'Beverages',

```

```

    'ProductName': 'Lakkalik\u00f6\u00f6ri',
    'QuantityPerUnit': '500 ml',
    'UnitsInStock': 57,
    'Discontinued': true
  },
  {
    'CategoryName': 'Beverages',
    'ProductName': 'Laughing Lumberjack Lager',
    'QuantityPerUnit': '24 - 12 oz bottles',
    'UnitsInStock': 52,
    'Discontinued': false
  },
  {
    'CategoryName': 'Beverages',
    'ProductName': 'Outback Lager',
    'QuantityPerUnit': '24 - 355 ml bottles',
    'UnitsInStock': 15,
    'Discontinued': false
  },
  {
    'CategoryName': 'Beverages',
    'ProductName': 'Rh\u00f6nbr\u00e4 Klosterbier',
    'QuantityPerUnit': '24 - 0.5 l bottles',
    'UnitsInStock': 125,
    'Discontinued': false
  },
  {
    'CategoryName': 'Beverages',
    'ProductName': 'Sasquatch Ale',
    'QuantityPerUnit': '24 - 12 oz bottles',
    'UnitsInStock': 111,
    'Discontinued': true
  },
  {
    'CategoryName': 'Beverages',
    'ProductName': 'Steeleye Stout',
    'QuantityPerUnit': '24 - 12 oz bottles',
    'UnitsInStock': 20,
    'Discontinued': true
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Aniseed Syrup',
    'QuantityPerUnit': '12 - 550 ml bottles',
    'UnitsInStock': 13,
    'Discontinued': true
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Chef Anton\'s Cajun Seasoning',
    'QuantityPerUnit': '48 - 6 oz jars',
    'UnitsInStock': 53,
    'Discontinued': false
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Genen Shouyu',

```

```

    'QuantityPerUnit': '24 - 250 ml bottles',
    'UnitsInStock': 39,
    'Discontinued': false
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Grandma\'s Boysenberry Spread',
    'QuantityPerUnit': '12 - 8 oz jars',
    'UnitsInStock': 120,
    'Discontinued': false
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Gula Malacca',
    'QuantityPerUnit': '20 - 2 kg bags',
    'UnitsInStock': 27,
    'Discontinued': true
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Louisiana Fiery Hot Pepper Sauce',
    'QuantityPerUnit': '32 - 8 oz bottles',
    'UnitsInStock': 76,
    'Discontinued': false
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Louisiana Hot Spiced Okra',
    'QuantityPerUnit': '24 - 8 oz jars',
    'UnitsInStock': 4,
    'Discontinued': false
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Northwoods Cranberry Sauce',
    'QuantityPerUnit': '12 - 12 oz jars',
    'UnitsInStock': 6,
    'Discontinued': true
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Original Frankfurter gr\u00fcne So\u00dfe',
    'QuantityPerUnit': '12 boxes',
    'UnitsInStock': 32,
    'Discontinued': false
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Sirop d'\u00e9rable',
    'QuantityPerUnit': '24 - 500 ml bottles',
    'UnitsInStock': 113,
    'Discontinued': false
  },
  {
    'CategoryName': 'Condiments',
    'ProductName': 'Vegie-spread',
    'QuantityPerUnit': '15 - 625 g jars',

```

```

        'UnitsInStock': 24,
        'Discontinued': false
    },
    {
        'CategoryName': 'Confections',
        'ProductName': 'Chocolade',
        'QuantityPerUnit': '10 pkgs.',
        'UnitsInStock': 15,
        'Discontinued': true
    },
    {
        'CategoryName': 'Confections',
        'ProductName': 'Gumb\u00e4r Gummib\u00e4rchen',
        'QuantityPerUnit': '100 - 250 g bags',
        'UnitsInStock': 15,
        'Discontinued': false
    },
    {
        'CategoryName': 'Confections',
        'ProductName': 'Maxilaku',
        'QuantityPerUnit': '24 - 50 g pkgs.',
        'UnitsInStock': 10,
        'Discontinued': false
    },
    {
        'CategoryName': 'Confections',
        'ProductName': 'NuNuCa Nu\u00df-Nougat-Creme',
        'QuantityPerUnit': '20 - 450 g glasses',
        'UnitsInStock': 76,
        'Discontinued': true
    },
    {
        'CategoryName': 'Confections',
        'ProductName': 'Pavlova',
        'QuantityPerUnit': '32 - 500 g boxes',
        'UnitsInStock': 29,
        'Discontinued': true
    },
    {
        'CategoryName': 'Confections',
        'ProductName': 'Schoggi Schokolade',
        'QuantityPerUnit': '100 - 100 g pieces',
        'UnitsInStock': 49,
        'Discontinued': true
    },
    {
        'CategoryName': 'Confections',
        'ProductName': 'Scottish Longbreads',
        'QuantityPerUnit': '10 boxes x 8 pieces',
        'UnitsInStock': 6,
        'Discontinued': true
    },
    {
        'CategoryName': 'Confections',
        'ProductName': 'Sir Rodney\'s Marmalade',
        'QuantityPerUnit': '30 gift boxes',
        'UnitsInStock': 40,

```



```

    'Discontinued': true
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'Sir Rodney\'s Scones',
    'QuantityPerUnit': '24 pkgs. x 4 pieces',
    'UnitsInStock': 3,
    'Discontinued': true
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'Tarte au sucre',
    'QuantityPerUnit': '48 pies',
    'UnitsInStock': 17,
    'Discontinued': false
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'Teatime Chocolate Biscuits',
    'QuantityPerUnit': '10 boxes x 12 pieces',
    'UnitsInStock': 25,
    'Discontinued': false
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'Valkoinen suklaa',
    'QuantityPerUnit': '12 - 100 g bars',
    'UnitsInStock': 65,
    'Discontinued': false
  },
  {
    'CategoryName': 'Confections',
    'ProductName': 'Zaanse koeken',
    'QuantityPerUnit': '10 - 4 oz boxes',
    'UnitsInStock': 36,
    'Discontinued': true
  },
  {
    'CategoryName': 'Dairy Products',
    'ProductName': 'Camembert Pierrot',
    'QuantityPerUnit': '15 - 300 g rounds',
    'UnitsInStock': 19,
    'Discontinued': false
  },
  {
    'CategoryName': 'Dairy Products',
    'ProductName': 'Flotemysost',
    'QuantityPerUnit': '10 - 500 g pkgs.',
    'UnitsInStock': 26,
    'Discontinued': false
  },
  {
    'CategoryName': 'Dairy Products',
    'ProductName': 'Geitost',
    'QuantityPerUnit': '500 g',
    'UnitsInStock': 112,
    'Discontinued': false
  }

```

```

    },
    {
      'CategoryName': 'Dairy Products',
      'ProductName': 'Gorgonzola Telino',
      'QuantityPerUnit': '12 - 100 g pkgs',
      'UnitsInStock': 0,
      'Discontinued': true
    },
    {
      'CategoryName': 'Dairy Products',
      'ProductName': 'Gudbrandsdalsost',
      'QuantityPerUnit': '10 kg pkg.',
      'UnitsInStock': 26,
      'Discontinued': true
    },
    {
      'CategoryName': 'Dairy Products',
      'ProductName': 'Mascarpone Fabioli',
      'QuantityPerUnit': '24 - 200 g pkgs.',
      'UnitsInStock': 9,
      'Discontinued': true
    },
    {
      'CategoryName': 'Dairy Products',
      'ProductName': 'Mozzarella di Giovanni',
      'QuantityPerUnit': '24 - 200 g pkgs.',
      'UnitsInStock': 14,
      'Discontinued': true
    },
    {
      'CategoryName': 'Dairy Products',
      'ProductName': 'Queso Cabrales',
      'QuantityPerUnit': '1 kg pkg.',
      'UnitsInStock': 22,
      'Discontinued': true
    },
    {
      'CategoryName': 'Dairy Products',
      'ProductName': 'Queso Manchego La Pastora',
      'QuantityPerUnit': '10 - 500 g pkgs.',
      'UnitsInStock': 86,
      'Discontinued': true
    },
    {
      'CategoryName': 'Dairy Products',
      'ProductName': 'Raclette Courdavault',
      'QuantityPerUnit': '5 kg pkg.',
      'UnitsInStock': 79,
      'Discontinued': false
    },
    {
      'CategoryName': 'Grains/Cereals',
      'ProductName': 'Filo Mix',
      'QuantityPerUnit': '16 - 2 kg boxes',
      'UnitsInStock': 38,
      'Discontinued': false
    },
  ],

```

```

{
  'CategoryName': 'Grains/Cereals',
  'ProductName': 'Gnocchi di nonna Alice',
  'QuantityPerUnit': '24 - 250 g pkgs.',
  'UnitsInStock': 21,
  'Discontinued': false
},
{
  'CategoryName': 'Grains/Cereals',
  'ProductName': 'Gustaf\'s Kn\u00e4ckebr\u00f6d',
  'QuantityPerUnit': '24 - 500 g pkgs.',
  'UnitsInStock': 104,
  'Discontinued': true
},
{
  'CategoryName': 'Grains/Cereals',
  'ProductName': 'Ravioli Angelo',
  'QuantityPerUnit': '24 - 250 g pkgs.',
  'UnitsInStock': 36,
  'Discontinued': true
},
{
  'CategoryName': 'Grains/Cereals',
  'ProductName': 'Tunnbr\u00f6d',
  'QuantityPerUnit': '12 - 250 g pkgs.',
  'UnitsInStock': 61,
  'Discontinued': true
},
{
  'CategoryName': 'Grains/Cereals',
  'ProductName': 'Wimmers gute Semmelkn\u00f6del',
  'QuantityPerUnit': '20 bags x 4 pieces',
  'UnitsInStock': 22,
  'Discontinued': true
},
{
  'CategoryName': 'Meat/Poultry',
  'ProductName': 'P\u00e4t\u00e9 chinois',
  'QuantityPerUnit': '24 boxes x 2 pies',
  'UnitsInStock': 115,
  'Discontinued': false
},
{
  'CategoryName': 'Meat/Poultry',
  'ProductName': 'Tourti\u00e8re',
  'QuantityPerUnit': '16 pies',
  'UnitsInStock': 21,
  'Discontinued': false
},
{
  'CategoryName': 'Produce',
  'ProductName': 'Longlife Tofu',
  'QuantityPerUnit': '5 kg pkg.',
  'UnitsInStock': 4,
  'Discontinued': false
},
{

```

```

    'CategoryName': 'Produce',
    'ProductName': 'Manjimup Dried Apples',
    'QuantityPerUnit': '50 - 300 g pkgs.',
    'UnitsInStock': 20,
    'Discontinued': false
  },
  {
    'CategoryName': 'Produce',
    'ProductName': 'Tofu',
    'QuantityPerUnit': '40 - 100 g pkgs.',
    'UnitsInStock': 35,
    'Discontinued': true
  },
  {
    'CategoryName': 'Produce',
    'ProductName': 'Uncle Bob\'s Organic Dried Pears',
    'QuantityPerUnit': '12 - 1 lb pkgs.',
    'UnitsInStock': 15,
    'Discontinued': true
  },
  {
    'CategoryName': 'Seafood',
    'ProductName': 'Boston Crab Meat',
    'QuantityPerUnit': '24 - 4 oz tins',
    'UnitsInStock': 123,
    'Discontinued': true
  },
  {
    'CategoryName': 'Seafood',
    'ProductName': 'Carnarvon Tigers',
    'QuantityPerUnit': '16 kg pkg.',
    'UnitsInStock': 42,
    'Discontinued': true
  },
  {
    'CategoryName': 'Seafood',
    'ProductName': 'Escargots de Bourgogne',
    'QuantityPerUnit': '24 pieces',
    'UnitsInStock': 62,
    'Discontinued': true
  },
  {
    'CategoryName': 'Seafood',
    'ProductName': 'Gravad lax',
    'QuantityPerUnit': '12 - 500 g pkgs.',
    'UnitsInStock': 11,
    'Discontinued': true
  },
  {
    'CategoryName': 'Seafood',
    'ProductName': 'Ikura',
    'QuantityPerUnit': '12 - 200 ml jars',
    'UnitsInStock': 31,
    'Discontinued': false
  },
  {
    'CategoryName': 'Seafood',

```

```

        'ProductName': 'Inlagd Sill',
        'QuantityPerUnit': '24 - 250 g jars',
        'UnitsInStock': 112,
        'Discontinued': false
    },
    {
        'CategoryName': 'Seafood',
        'ProductName': 'Jack\'s New England Clam Chowder',
        'QuantityPerUnit': '12 - 12 oz cans',
        'UnitsInStock': 85,
        'Discontinued': false
    },
    {
        'CategoryName': 'Seafood',
        'ProductName': 'Konbu',
        'QuantityPerUnit': '2 kg box',
        'UnitsInStock': 24,
        'Discontinued': false
    },
    {
        'CategoryName': 'Seafood',
        'ProductName': 'Nord-Ost Matjeshering',
        'QuantityPerUnit': '10 - 200 g glasses',
        'UnitsInStock': 10,
        'Discontinued': false
    },
    {
        'CategoryName': 'Seafood',
        'ProductName': 'R\u00f6d Kaviar',
        'QuantityPerUnit': '24 - 150 g jars',
        'UnitsInStock': 101,
        'Discontinued': false
    },
    {
        'CategoryName': 'Seafood',
        'ProductName': 'Rogede sild',
        'QuantityPerUnit': '1k pkg.',
        'UnitsInStock': 5,
        'Discontinued': false
    },
    {
        'CategoryName': 'Seafood',
        'ProductName': 'Spegesild',
        'QuantityPerUnit': '4 - 450 g glasses',
        'UnitsInStock': 95,
        'Discontinued': true
    }
];
export const customerData: Object[] = [
    {
        'CustomerID': 'ALFKI',
        'ContactName': 'Maria ',
        'CompanyName': 'Alfreds Futterkiste',
        'Address': 'Obere Str. 57',
        'Country': 'Germany'
    },
    {

```

```

    'CustomerID': 'ANATR',
    'ContactName': 'Ana Trujillo',
    'CompanyName': 'Ana Trujillo Emparedados y helados',
    'Address': 'Avda. de la Constitución 2222',
    'Country': 'Mexico'
  },
  {
    'CustomerID': 'ANTON',
    'ContactName': 'Antonio Moreno',
    'CompanyName': 'Antonio Moreno Taquería',
    'Address': 'Mataderos 2312',
    'Country': 'Mexico'
  },
  {
    'CustomerID': 'AROUT',
    'ContactName': 'Thomas Hardy',
    'CompanyName': 'Around the Horn',
    'Address': '120 Hanover Sq.',
    'Country': 'UK'
  },
  {
    'CustomerID': 'BERGS',
    'ContactName': 'Christina Berglund',
    'CompanyName': 'Berglunds snabbköp',
    'Address': 'Berguvsvägen 8',
    'Country': 'Sweden'
  },
  {
    'CustomerID': 'BLAUS',
    'ContactName': 'Hanna Moos',
    'CompanyName': 'Blauer See Delikatessen',
    'Address': 'Forsterstr. 57',
    'Country': 'Germany'
  },
  {
    'CustomerID': 'BLONP',
    'ContactName': 'Frédérique Citeaux',
    'CompanyName': 'Blondesddsl père et fils',
    'Address': '24, place Kléber',
    'Country': 'France'
  },
  {
    'CustomerID': 'BOLID',
    'ContactName': 'Martín Sommer',
    'CompanyName': 'Bólido Comidas preparadas',
    'Address': 'C/ Araquil, 67',
    'Country': 'Spain'
  },
  {
    'CustomerID': 'BONAP',
    'ContactName': 'Laurence Lebihan',
    'CompanyName': 'Bon app',
    'Address': '12, rue des Bouchers',
    'Country': 'France'
  },
  {
    'CustomerID': 'BOTTM',

```

```

    'ContactName': 'Elizabeth Lincoln',
    'CompanyName': 'Bottom-Dollar Markets',
    'Address': '23 Tsawassen Blvd.',
    'Country': 'Canada'
  },
  {
    'CustomerID': 'BSBEV',
    'ContactName': 'Victoria Ashworth',
    'CompanyName': 'B\'s Beverages',
    'Address': 'Fauntleroy Circus',
    'Country': 'UK'
  },
  {
    'CustomerID': 'CACTU',
    'ContactName': 'Patricio Simpson',
    'CompanyName': 'Cactus Comidas para llevar',
    'Address': 'Cerrito 333',
    'Country': 'Argentina'
  },
  {
    'CustomerID': 'CENTC',
    'ContactName': 'Francisco Chang',
    'CompanyName': 'Centro comercial Moctezuma',
    'Address': 'Sierras de Granada 9993',
    'Country': 'Mexico'
  },
  {
    'CustomerID': 'CHOPS',
    'ContactName': 'Yang Wang',
    'CompanyName': 'Chop-suey Chinese',
    'Address': 'Hauptstr. 29',
    'Country': 'Switzerland'
  },
  {
    'CustomerID': 'COMMI',
    'ContactName': 'Pedro Afonso',
    'CompanyName': 'Comércio Mineiro',
    'Address': 'Av. dos Lusíadas, 23',
    'Country': 'Brazil'
  },
  {
    'CustomerID': 'CONSH',
    'ContactName': 'Elizabeth Brown',
    'CompanyName': 'Consolidated Holdings',
    'Address': 'Berkeley Gardens 12 Brewery',
    'Country': 'UK'
  },
  {
    'CustomerID': 'DRACD',
    'ContactName': 'Sven Ottlieb',
    'CompanyName': 'Drachenblut Delikatessen',
    'Address': 'Walserweg 21',
    'Country': 'Germany'
  },
  {
    'CustomerID': 'DUMON',
    'ContactName': 'Janine Labrune',

```

```

    'CompanyName': 'Du monde entier',
    'Address': '67, rue des Cinquante Otages',
    'Country': 'France'
  },
  {
    'CustomerID': 'EASTC',
    'ContactName': 'Ann Devon',
    'CompanyName': 'Eastern Connection',
    'Address': '35 King George',
    'Country': 'UK'
  },
  {
    'CustomerID': 'ERNSH',
    'ContactName': 'Roland Mendel',
    'CompanyName': 'Ernst Handel',
    'Address': 'Kirchgasse 6',
    'Country': 'Austria'
  },
  {
    'CustomerID': 'FAMIA',
    'ContactName': 'Aria Cruz',
    'CompanyName': 'Familia Arquibaldo',
    'Address': 'Rua Orós, 92',
    'Country': 'Brazil'
  },
  {
    'CustomerID': 'FISSA',
    'ContactName': 'Diego Roel',
    'CompanyName': 'FISSA Fabrica Inter. Salchichas S.A.',
    'Address': 'C/ Moralzarzal, 86',
    'Country': 'Spain'
  },
  {
    'CustomerID': 'FOLIG',
    'ContactName': 'Martine Rancé',
    'CompanyName': 'Folies gourmandes',
    'Address': '184, chaussée de Tournai',
    'Country': 'France'
  },
  {
    'CustomerID': 'FOLKO',
    'ContactName': 'Maria Larsson',
    'CompanyName': 'Folk och fä HB',
    'Address': 'Åkergatan 24',
    'Country': 'Sweden'
  },
  {
    'CustomerID': 'FRANK',
    'ContactName': 'Peter Franken',
    'CompanyName': 'Frankenversand',
    'Address': 'Berliner Platz 43',
    'Country': 'Germany'
  },
  {
    'CustomerID': 'FRANR',
    'ContactName': 'Carine Schmitt',
    'CompanyName': 'France restauration',

```



```

    'Address': '54, rue Royale',
    'Country': 'France'
  },
  {
    'CustomerID': 'FRANS',
    'ContactName': 'Paolo Accorti',
    'CompanyName': 'Franchi S.p.A.',
    'Address': 'Via Monte Bianco 34',
    'Country': 'Italy'
  },
  {
    'CustomerID': 'FURIB',
    'ContactName': 'Lino Rodriguez',
    'CompanyName': 'Furia Bacalhau e Frutos do Mar',
    'Address': 'Jardim das rosas n. 32',
    'Country': 'Portugal'
  },
  {
    'CustomerID': 'GALED',
    'ContactName': 'Eduardo Saavedra',
    'CompanyName': 'Galería del gastrónomo',
    'Address': 'Rambla de Cataluña, 23',
    'Country': 'Spain'
  },
  {
    'CustomerID': 'GODOS',
    'ContactName': 'José Pedro Freyre',
    'CompanyName': 'Godos Cocina Típica',
    'Address': 'C/ Romero, 33',
    'Country': 'Spain'
  },
  {
    'CustomerID': 'GOURL',
    'ContactName': 'André Fonseca',
    'CompanyName': 'Gourmet Lanchonetes',
    'Address': 'Av. Brasil, 442',
    'Country': 'Brazil'
  },
  {
    'CustomerID': 'GREAL',
    'ContactName': 'Howard Snyder',
    'CompanyName': 'Great Lakes Food Market',
    'Address': '2732 Baker Blvd.',
    'Country': 'USA'
  },
  {
    'CustomerID': 'GROSR',
    'ContactName': 'Manuel Pereira',
    'CompanyName': 'GROSELLA-Restaurante',
    'Address': '5ª Ave. Los Palos Grandes',
    'Country': 'Venezuela'
  },
  {
    'CustomerID': 'HANAR',
    'ContactName': 'Mario Pontes',
    'CompanyName': 'Hanari Carnes',
    'Address': 'Rua do Paço, 67',

```

```

    'Country': 'Brazil'
  },
  {
    'CustomerID': 'HILAA',
    'ContactName': 'Carlos Hernández',
    'CompanyName': 'HILARION-Abastos',
    'Address': 'Carrera 22 con Ave. Carlos Soublette #8-35',
    'Country': 'Venezuela'
  },
  {
    'CustomerID': 'HUNGC',
    'ContactName': 'Yoshi Latimer',
    'CompanyName': 'Hungry Coyote Import Store',
    'Address': 'City Center Plaza 516 Main St.',
    'Country': 'USA'
  },
  {
    'CustomerID': 'HUNGO',
    'ContactName': 'Patricia McKenna',
    'CompanyName': 'Hungry Owl All-Night Grocers',
    'Address': '8 Johnstown Road',
    'Country': 'Ireland'
  },
  {
    'CustomerID': 'ISLAT',
    'ContactName': 'Helen Bennett',
    'CompanyName': 'Island Trading',
    'Address': 'Garden House Crowther Way',
    'Country': 'UK'
  },
  {
    'CustomerID': 'KOENE',
    'ContactName': 'Philip Cramer',
    'CompanyName': 'Königlich Essen',
    'Address': 'Maubelstr. 90',
    'Country': 'Germany'
  },
  {
    'CustomerID': 'LACOR',
    'ContactName': 'Daniel Tonini',
    'CompanyName': 'La corne d\'abondance',
    'Address': '67, avenue de l\'Europe',
    'Country': 'France'
  },
  {
    'CustomerID': 'LAMAI',
    'ContactName': 'Annette Roulet',
    'CompanyName': 'La maison d\'Asie',
    'Address': '1 rue Alsace-Lorraine',
    'Country': 'France'
  },
  {
    'CustomerID': 'LAUGB',
    'ContactName': 'Yoshi Tannamuri',
    'CompanyName': 'Laughing Bacchus Wine Cellars',
    'Address': '1900 Oak St.',
    'Country': 'Canada'
  }

```

```

},
{
    'CustomerID': 'LAZYK',
    'ContactName': 'John Steel',
    'CompanyName': 'Lazy K Kountry Store',
    'Address': '12 Orchestra Terrace',
    'Country': 'USA'
},
{
    'CustomerID': 'LEHMS',
    'ContactName': 'Renate Messner',
    'CompanyName': 'Lehmanns Marktstand',
    'Address': 'Magazinweg 7',
    'Country': 'Germany'
},
{
    'CustomerID': 'LETSS',
    'ContactName': 'Jaime Yorres',
    'CompanyName': 'Let\'s Stop N Shop',
    'Address': '87 Polk St. Suite 5',
    'Country': 'USA'
},
{
    'CustomerID': 'LILAS',
    'ContactName': 'Carlos González',
    'CompanyName': 'LILA-Supermercado',
    'Address': 'Carrera 52 con Ave. Bolívar #65-98 Llano Largo',
    'Country': 'Venezuela'
},
{
    'CustomerID': 'LINOD',
    'ContactName': 'Felipe Izquierdo',
    'CompanyName': 'LINO-Delicateses',
    'Address': 'Ave. 5 de Mayo Porlamar',
    'Country': 'Venezuela'
},
{
    'CustomerID': 'LONEP',
    'ContactName': 'Fran Wilson',
    'CompanyName': 'Lonesome Pine Restaurant',
    'Address': '89 Chiaroscuro Rd.',
    'Country': 'USA'
},
{
    'CustomerID': 'MAGAA',
    'ContactName': 'Giovanni Rovelli',
    'CompanyName': 'Magazzini Alimentari Riuniti',
    'Address': 'Via Ludovico il Moro 22',
    'Country': 'Italy'
},
{
    'CustomerID': 'MAISD',
    'ContactName': 'Catherine Dewey',
    'CompanyName': 'Maison Dewey',
    'Address': 'Rue Joseph-Bens 532',
    'Country': 'Belgium'
},
},

```

```

{
    'CustomerID': 'MEREPA',
    'ContactName': 'Jean Fresnière',
    'CompanyName': 'Mère Paillarde',
    'Address': '43 rue St. Laurent',
    'Country': 'Canada'
},
{
    'CustomerID': 'MORGK',
    'ContactName': 'Alexander Feuer',
    'CompanyName': 'Morgenstern Gesundkost',
    'Address': 'Heerstr. 22',
    'Country': 'Germany'
},
{
    'CustomerID': 'NORTS',
    'ContactName': 'Simon Crowther',
    'CompanyName': 'North/South',
    'Address': 'South House 300 Queensbridge',
    'Country': 'UK'
},
{
    'CustomerID': 'OCEAN',
    'ContactName': 'Yvonne Moncada',
    'CompanyName': 'Océano Atlántico Ltda.',
    'Address': 'Ing. Gustavo Moncada 8585 Piso 20-A',
    'Country': 'Argentina'
},
{
    'CustomerID': 'OLDWO',
    'ContactName': 'Rene Phillips',
    'CompanyName': 'Old World Delicatessen',
    'Address': '2743 Bering St.',
    'Country': 'USA'
},
{
    'CustomerID': 'OTTIK',
    'ContactName': 'Henriette Pfalzheim',
    'CompanyName': 'Ottilies Käseladen',
    'Address': 'Mehrheimerstr. 369',
    'Country': 'Germany'
},
{
    'CustomerID': 'PARIS',
    'ContactName': 'Marie Bertrand',
    'CompanyName': 'Paris spécialités',
    'Address': '265, boulevard Charonne',
    'Country': 'France'
},
{
    'CustomerID': 'PERIC',
    'ContactName': 'Guillermo Fernández',
    'CompanyName': 'Pericles Comidas clásicas',
    'Address': 'Calle Dr. Jorge Cash 321',
    'Country': 'Mexico'
},
{

```

```

    'CustomerID': 'PICCO',
    'ContactName': 'Georg Pipps',
    'CompanyName': 'Piccolo und mehr',
    'Address': 'Geislweg 14',
    'Country': 'Austria'
  },
  {
    'CustomerID': 'PRINI',
    'ContactName': 'Isabel de Castro',
    'CompanyName': 'Princesa Isabel Vinhos',
    'Address': 'Estrada da saúde n. 58',
    'Country': 'Portugal'
  },
  {
    'CustomerID': 'QUEDE',
    'ContactName': 'Bernardo Batista',
    'CompanyName': 'Que Delícia',
    'Address': 'Rua da Panificadora, 12',
    'Country': 'Brazil'
  },
  {
    'CustomerID': 'QUEEN',
    'ContactName': 'Lúcia Carvalho',
    'CompanyName': 'Queen Cozinha',
    'Address': 'Alameda dos Canários, 891',
    'Country': 'Brazil'
  },
  {
    'CustomerID': 'QUICK',
    'ContactName': 'Horst Kloss',
    'CompanyName': 'QUICK-Stop',
    'Address': 'Taucherstraße 10',
    'Country': 'Germany'
  },
  {
    'CustomerID': 'RANCH',
    'ContactName': 'Sergio Gutiérrez',
    'CompanyName': 'Rancho grande',
    'Address': 'Av. del Libertador 900',
    'Country': 'Argentina'
  },
  {
    'CustomerID': 'RATTC',
    'ContactName': 'Paula Wilson',
    'CompanyName': 'Rattlesnake Canyon Grocery',
    'Address': '2817 Milton Dr.',
    'Country': 'USA'
  },
  {
    'CustomerID': 'REGGC',
    'ContactName': 'Maurizio Moroni',
    'CompanyName': 'Reggiani Caseifici',
    'Address': 'Strada Provinciale 124',
    'Country': 'Italy'
  },
  {
    'CustomerID': 'RICAR',

```

```

    'ContactName': 'Janete Limeira',
    'CompanyName': 'Ricardo Adocicados',
    'Address': 'Av. Copacabana, 267',
    'Country': 'Brazil'
  },
  {
    'CustomerID': 'RICSU',
    'ContactName': 'Michael Holz',
    'CompanyName': 'Richter Supermarkt',
    'Address': 'Grenzacherweg 237',
    'Country': 'Switzerland'
  },
  {
    'CustomerID': 'ROMEY',
    'ContactName': 'Alejandra Camino',
    'CompanyName': 'Romero y tomillo',
    'Address': 'Gran Vía, 1',
    'Country': 'Spain'
  },
  {
    'CustomerID': 'SANTG',
    'ContactName': 'Jonas Bergulfsen',
    'CompanyName': 'Santé Gourmet',
    'Address': 'Erling Skakkes gate 78',
    'Country': 'Norway'
  },
  {
    'CustomerID': 'SAVEA',
    'ContactName': 'Jose Pavarotti',
    'CompanyName': 'Save-a-lot Markets',
    'Address': '187 Suffolk Ln.',
    'Country': 'USA'
  },
  {
    'CustomerID': 'SEVES',
    'ContactName': 'Hari Kumar',
    'CompanyName': 'Seven Seas Imports',
    'Address': '90 Wadhurst Rd.',
    'Country': 'UK'
  },
  {
    'CustomerID': 'SIMOB',
    'ContactName': 'Jytte Petersen',
    'CompanyName': 'Simons bistro',
    'Address': 'Vinbæltet 34',
    'Country': 'Denmark'
  },
  {
    'CustomerID': 'SPECB',
    'ContactName': 'Dominique Perrier',
    'CompanyName': 'Spécialités du monde',
    'Address': '25, rue Lauriston',
    'Country': 'France'
  },
  {
    'CustomerID': 'SPLIR',
    'ContactName': 'Art Braunschweiger',

```

```

    'CompanyName': 'Split Rail Beer & Ale',
    'Address': 'P.O. Box 555',
    'Country': 'USA'
  },
  {
    'CustomerID': 'SUPRD',
    'ContactName': 'Pascale Cartrain',
    'CompanyName': 'Suprêmes délices',
    'Address': 'Boulevard Tirou, 255',
    'Country': 'Belgium'
  },
  {
    'CustomerID': 'THEBI',
    'ContactName': 'Liz Nixon',
    'CompanyName': 'The Big Cheese',
    'Address': '89 Jefferson Way Suite 2',
    'Country': 'USA'
  },
  {
    'CustomerID': 'THECR',
    'ContactName': 'Liu Wong',
    'CompanyName': 'The Cracker Box',
    'Address': '55 Grizzly Peak Rd.',
    'Country': 'USA'
  },
  {
    'CustomerID': 'TOMSP',
    'ContactName': 'Karin Josephs',
    'CompanyName': 'Toms Spezialitäten',
    'Address': 'Luisenstr. 48',
    'Country': 'Germany'
  },
  {
    'CustomerID': 'TORTU',
    'ContactName': 'Miguel Angel Paolino',
    'CompanyName': 'Tortuga Restaurante',
    'Address': 'Avda. Azteca 123',
    'Country': 'Mexico'
  },
  {
    'CustomerID': 'TRADH',
    'ContactName': 'Anabela Domingues',
    'CompanyName': 'Tradição Hipermercados',
    'Address': 'Av. Inês de Castro, 414',
    'Country': 'Brazil'
  },
  {
    'CustomerID': 'TRAIH',
    'ContactName': 'Helvetius Nagy',
    'CompanyName': 'Trail\'s Head Gourmet Provisioners',
    'Address': '722 DaVinci Blvd.',
    'Country': 'USA'
  },
  {
    'CustomerID': 'VAFFE',
    'ContactName': 'Palle Ibsen',
    'CompanyName': 'Vaffeljernet',

```

```

    'Address': 'Smagsloget 45',
    'Country': 'Denmark'
  },
  {
    'CustomerID': 'VICTE',
    'ContactName': 'Mary Saveley',
    'CompanyName': 'Victuailles en stock',
    'Address': '2, rue du Commerce',
    'Country': 'France'
  },
  {
    'CustomerID': 'VINET',
    'ContactName': 'Paul Henriot',
    'CompanyName': 'Vins et alcools Chevalier',
    'Address': '59 rue de l\'Abbaye',
    'Country': 'France'
  },
  {
    'CustomerID': 'WANDK',
    'ContactName': 'Rita Müller',
    'CompanyName': 'Die Wandernde Kuh',
    'Address': 'Adenauerallee 900',
    'Country': 'Germany'
  },
  {
    'CustomerID': 'WARTH',
    'ContactName': 'Pirkko Koskitalo',
    'CompanyName': 'Wartian Herkku',
    'Address': 'Torikatu 38',
    'Country': 'Finland'
  },
  {
    'CustomerID': 'WELLI',
    'ContactName': 'Paula Parente',
    'CompanyName': 'Wellington Importadora',
    'Address': 'Rua do Mercado, 12',
    'Country': 'Brazil'
  },
  {
    'CustomerID': 'WHITC',
    'ContactName': 'Karl Jablonski',
    'CompanyName': 'White Clover Markets',
    'Address': '305 - 14th Ave. S. Suite 3B',
    'Country': 'USA'
  },
  {
    'CustomerID': 'WILMK',
    'ContactName': 'Matti Karttunen',
    'CompanyName': 'Wilman Kala',
    'Address': 'Keskuskatu 45',
    'Country': 'Finland'
  },
  {
    'CustomerID': 'WOLZA',
    'ContactName': 'Zbyszek Piestrzeniewicz',
    'CompanyName': 'Wolski Zajazd',
    'Address': 'ul. Filtrowa 68',

```



```

        'Country': 'Poland'
    }
];
type cType = { CustomerID: string, ContactName: string, CustomerName: string };
export const data: Object[] = orderData.map((item: cType) => {
    let name: cType = (<cType[]>customerData).filter((cItem: cType) => {
        return cItem.CustomerID === item.CustomerID;
    })[0];
    item.CustomerName = (name || <cType>{}).ContactName;
    return item;
});
export const inventoryData: Object[] = [
    {
        'Inventor': 'Kia Silverbrook',
        'NumberofPatentFamilies': 4737,
        'Country': 'Australia',
        'Number of INPADOC patents': 9839,
        'Active': '1994-2016',
        'Mainfieldsofinvention': 'Printing, Digital paper, Internet,
Electronics, Lab-on-a-chip, MEMS, Mechanical, VLSI',
    },
    {
        'Inventor': 'Shunpei Yamazaki',
        'NumberofPatentFamilies': 4677,
        'Country': 'Japan',
        'Number of INPADOC patents': '10000+',
        'Active': '1976-2016',
        'Mainfieldsofinvention': 'Thin film transistors, Liquid crystal
displays, Solar cells, Flash memory, OLED',
    },
    {
        'Inventor': 'Lowell L. Wood, Jr.',
        'NumberofPatentFamilies': 1419,
        'Country': 'USA',
        'Number of INPADOC patents': 1332,
        'Active': '1977-2016',
        'Mainfieldsofinvention': 'Mosquito laser, Nuclear weapons',
    },
    {
        'Inventor': 'Paul Lapstun',
        'NumberofPatentFamilies': 1281,
        'Country': 'Australia',
        'Number of INPADOC patents': 3099,
        'Active': '2000-2016',
        'Mainfieldsofinvention': 'Printing, Digital paper, Internet,
Electronics, CGI, VLSI',
    },
    {
        'Inventor': 'Gurtej Sandhu',
        'NumberofPatentFamilies': 1255,
        'Country': 'India',
        'Number of INPADOC patents': 2038,
        'Active': '1991-2016',
        'Mainfieldsofinvention': 'Thin film processes and materials, VLSI,
Semiconductor device fabrication',
    },
],

```

```

{
  'Inventor': 'Jun Koyama',
  'NumberofPatentFamilies': 1240,
  'Country': 'Japan',
  'Number of INPADOC patents': 4126,
  'Active': '1991-2016',
  'Mainfieldsofinvention': 'Thin film transistors, Liquid crystal
displays, OLED',
},
{
  'Inventor': 'Roderick A. Hyde',
  'NumberofPatentFamilies': 1240,
  'Country': 'USA',
  'Number of INPADOC patents': 3360,
  'Active': '2001-2016',
  'Mainfieldsofinvention': 'Various',
},
{
  'Inventor': 'Leonard Forbes',
  'NumberofPatentFamilies': 1093,
  'Country': 'Canada',
  'Number of INPADOC patents': 1398,
  'Active': '1991-2016',
  'Mainfieldsofinvention': 'Semiconductor Memories, CCDs, Thin film
processes and materials, VLSI',
},
{
  'Inventor': 'Thomas Edison',
  'NumberofPatentFamilies': 1084,
  'Country': 'USA',
  'Number of INPADOC patents': 2332,
  'Active': '1847(b)-1931(d)',
  'Mainfieldsofinvention': 'Electric power, Lighting, Batteries,
Phonograph, Cement, Telegraphy, Mining',
},
{
  'Inventor': 'Donald E. Weder',
  'NumberofPatentFamilies': 999,
  'Country': 'USA',
  'Number of INPADOC patents': 1993,
  'Active': '1976-2015',
  'Mainfieldsofinvention': 'Florist supplies',
},
{
  'Inventor': 'George Albert Lyon',
  'NumberofPatentFamilies': 993,
  'Country': 'Canada',
  'Number of INPADOC patents': 'NA',
  'Active': '1882(b)-1961(d)',
  'Mainfieldsofinvention': 'Automotive, Stainless steel products',
},
{
  'Inventor': 'John F. O\'Connor',
  'NumberofPatentFamilies': 949,
  'Country': 'USA',
  'Number of INPADOC patents': 'NA',
  'Active': '1864(b)-1938(d)',
}

```

```

    'Mainfieldsofinvention': 'Railway draft gearing',
  },
  {
    'Inventor': 'Melvin De Groote',
    'NumberofPatentFamilies': 925,
    'Country': 'USA',
    'Number of INPADOC patents': 'NA',
    'Active': '1895(b)-1963(d)',
    'Mainfieldsofinvention': 'Chemical de-emulsifiers',
  },
  {
    'Inventor': 'Jay S. Walker',
    'NumberofPatentFamilies': 918,
    'Country': 'USA',
    'Number of INPADOC patents': 2206,
    'Active': '1998-2016',
    'Mainfieldsofinvention': 'Gaming machines',
  },
  {
    'Inventor': 'Edward K. Y. Jung',
    'NumberofPatentFamilies': 911,
    'Country': 'USA',
    'Number of INPADOC patents': 2254,
    'Active': '1996-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Francis H. Richards',
    'NumberofPatentFamilies': 894,
    'Country': 'USA',
    'Number of INPADOC patents': 'NA',
    'Active': '1850(b)-19??(d)',
    'Mainfieldsofinvention': 'Mechanical, automation',
  },
  {
    'Inventor': 'Kangguo Cheng',
    'NumberofPatentFamilies': 884,
    'Country': 'USA',
    'Number of INPADOC patents': 1314,
    'Active': '2004-2016',
    'Mainfieldsofinvention': 'Semiconductor device fabrication,
Semiconductor memory, Semiconductor device',
  },
  {
    'Inventor': 'Clarence T. Tegreene',
    'NumberofPatentFamilies': 872,
    'Country': 'USA',
    'Number of INPADOC patents': 2255,
    'Active': '2000-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Ahmadreza Rofougaran',
    'NumberofPatentFamilies': 808,
    'Country': 'USA',
    'Number of INPADOC patents': 1396,
    'Active': '2002-2016',
  }

```

```

    'Mainfieldsofinvention': 'Radio Frequency Integrated Circuits',
  },
  {
    'Inventor': 'Shou-Shan Fan',
    'NumberofPatentFamilies': 805,
    'Country': 'China',
    'Number of INPADOC patents': 2120,
    'Active': '2006-2016',
    'Mainfieldsofinvention': 'Carbon nanotubes and applications of
carbon nanotubes',
  },
  {
    'Inventor': 'Michael J. Sullivan',
    'NumberofPatentFamilies': 788,
    'Country': 'USA',
    'Number of INPADOC patents': 1560,
    'Active': '1977-2016',
    'Mainfieldsofinvention': 'Golf balls',
  },
  {
    'Inventor': 'Rick Allen Hamilton II',
    'NumberofPatentFamilies': 773,
    'Country': 'USA',
    'Number of INPADOC patents': 1064,
    'Active': '1999-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Warren Farnworth',
    'NumberofPatentFamilies': 770,
    'Country': 'USA',
    'Number of INPADOC patents': 931,
    'Active': '1990-2016',
    'Mainfieldsofinvention': 'Semiconductor packaging',
  },
  {
    'Inventor': 'Carleton Ellis',
    'NumberofPatentFamilies': 753,
    'Country': 'USA',
    'Number of INPADOC patents': 'NA',
    'Active': '1876(b)-1941(d)',
    'Mainfieldsofinvention': 'Margarine, Polyester, Anti-knock gasoline,
Paint stripper',
  },
  {
    'Inventor': 'William H. Eby',
    'NumberofPatentFamilies': 733,
    'Country': 'USA',
    'Number of INPADOC patents': 758,
    'Active': '1994-2016',
    'Mainfieldsofinvention': 'Transgenic soybeans',
  },
  {
    'Inventor': 'Hideo Ando',
    'NumberofPatentFamilies': 728,
    'Country': 'Japan',
    'Number of INPADOC patents': 2588,
  }

```

```

    'Active': '1983-2016',
    'Mainfieldsofinvention': 'Optical recording',
  },
  {
    'Inventor': 'Salman Akram',
    'NumberofPatentFamilies': 728,
    'Country': 'USA',
    'Number of INPADOC patents': 915,
    'Active': '1995-2016',
    'Mainfieldsofinvention': 'Semiconductor packaging',
  },
  {
    'Inventor': 'George Spector',
    'NumberofPatentFamilies': 722,
    'Country': 'USA',
    'Number of INPADOC patents': 747,
    'Active': '1976-1998',
    'Mainfieldsofinvention': 'Gadgets, Toys',
  },
  {
    'Inventor': 'Jeyhan Karaoguz',
    'NumberofPatentFamilies': 721,
    'Country': 'USA',
    'Number of INPADOC patents': 1530,
    'Active': '1996-2016',
    'Mainfieldsofinvention': 'Wireless communications, Computer
networks',
  },
  {
    'Inventor': 'Elihu Thomson',
    'NumberofPatentFamilies': 696,
    'Country': 'UK',
    'Number of INPADOC patents': 'NA',
    'Active': '1853(b)-1937(d)',
    'Mainfieldsofinvention': 'Electric power, Arc lamp, Electric motors,
Lightning arrester, Arc welder',
  },
  {
    'Inventor': 'Austin L. Gurney',
    'NumberofPatentFamilies': 695,
    'Country': 'USA',
    'Number of INPADOC patents': 3909,
    'Active': '1999-2016',
    'Mainfieldsofinvention': 'Proteins, Antibodies',
  },
  {
    'Inventor': 'Tetsujiro Kondo',
    'NumberofPatentFamilies': 684,
    'Country': 'Japan',
    'Number of INPADOC patents': 4158,
    'Active': '1987-2015',
    'Mainfieldsofinvention': 'Signal processing, Image processing',
  },
  {
    'Inventor': 'Nathan Myhrvold',
    'NumberofPatentFamilies': 661,
    'Country': 'USA',

```

```

    'Number of INPADOC patents': 1690,
    'Active': '1994-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'William I. Wood',
    'NumberofPatentFamilies': 653,
    'Country': 'USA',
    'Number of INPADOC patents': 3560,
    'Active': '1981-2016',
    'Mainfieldsofinvention': 'Proteins, Antibodies',
  },
  {
    'Inventor': 'Simon R. Walmsley',
    'NumberofPatentFamilies': 651,
    'Country': 'Australia',
    'Number of INPADOC patents': 1249,
    'Active': '1995-2015',
    'Mainfieldsofinvention': 'Printing, Electronics, VLSI,
Cryptography',
  },
  {
    'Inventor': 'Mark Malamud',
    'NumberofPatentFamilies': 632,
    'Country': 'USA',
    'Number of INPADOC patents': 1759,
    'Active': '1997-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Royce A. Levien',
    'NumberofPatentFamilies': 630,
    'Country': 'USA',
    'Number of INPADOC patents': 1799,
    'Active': '1997-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Audrey D. Goddard',
    'NumberofPatentFamilies': 622,
    'Country': 'USA',
    'Number of INPADOC patents': 3416,
    'Active': '1997-2014',
    'Mainfieldsofinvention': 'Proteins, Antibodies',
  },
  {
    'Inventor': 'Muriel Y. Ishikawa',
    'NumberofPatentFamilies': 619,
    'Country': 'USA',
    'Number of INPADOC patents': 1660,
    'Active': '2002-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Robert W. Lord',
    'NumberofPatentFamilies': 618,
    'Country': 'USA',

```

```

    'Number of INPADOC patents': 1708,
    'Active': '2003-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Jerome Lemelson',
    'NumberofPatentFamilies': 606,
    'Country': 'USA',
    'Number of INPADOC patents': 'NA',
    'Active': '1923(b)-1997(d)',
    'Mainfieldsofinvention': 'Toys, Industrial robots, Cordless
telephones, Fax machines, Videocassette recorders',
  },
  {
    'Inventor': 'Béla Barényi',
    'NumberofPatentFamilies': 595,
    'Country': 'Austria',
    'Number of INPADOC patents': 1244,
    'Active': '1907(b)-1997(d)',
    'Mainfieldsofinvention': 'Passive safety in automobiles',
  },
  {
    'Inventor': 'Kie Y Ahn',
    'NumberofPatentFamilies': 593,
    'Country': 'USA',
    'Number of INPADOC patents': 709,
    'Active': '1976-2016',
    'Mainfieldsofinvention': 'Thin film processes and materials, VLSI,
Semiconductor device fabrication',
  },
  {
    'Inventor': 'Tadahiro Ohmi',
    'NumberofPatentFamilies': 592,
    'Country': 'Japan',
    'Number of INPADOC patents': 2691,
    'Active': '1981-2016',
    'Mainfieldsofinvention': 'Thin film processes and materials,
Semiconductor device fabrication',
  },
  {
    'Inventor': 'Jordin T. Kare',
    'NumberofPatentFamilies': 585,
    'Country': 'USA',
    'Number of INPADOC patents': 1559,
    'Active': '1992-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Paul J. Godowski',
    'NumberofPatentFamilies': 579,
    'Country': 'USA',
    'Number of INPADOC patents': 2605,
    'Active': '1994-2014',
    'Mainfieldsofinvention': 'Proteins, Antibodies',
  },
  {
    'Inventor': 'Artur Fischer',

```

```

    'NumberofPatentFamilies': 570,
    'Country': 'Germany',
    'Number of INPADOC patents': 3097,
    'Active': '1976-2002',
    'Mainfieldsofinvention': 'Fasteners, Construction toys',
  },
  {
    'Inventor': 'Edward J. Nowak',
    'NumberofPatentFamilies': 564,
    'Country': 'USA',
    'Number of INPADOC patents': 1145,
    'Active': '1979-2016',
    'Mainfieldsofinvention': 'Semiconductor device fabrication,
Semiconductor memory, Semiconductor device',
  },
  {
    'Inventor': 'Louis L. Hsu',
    'NumberofPatentFamilies': 551,
    'Country': 'USA',
    'Number of INPADOC patents': 914,
    'Active': '1988-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Edwin H. Land',
    'NumberofPatentFamilies': 535,
    'Country': 'USA',
    'Number of INPADOC patents': 1236,
    'Active': '1909(b)-1991(d)',
    'Mainfieldsofinvention': 'Instant photography, Polarizing film',
  },
  {
    'Inventor': 'Henri Dreyfus',
    'NumberofPatentFamilies': 524,
    'Country': 'Switzerland',
    'Number of INPADOC patents': 2117,
    'Active': '1882(b)-1944(d)',
    'Mainfieldsofinvention': 'Polymers, Synthetic fibers, Dyes',
  },
  {
    'Inventor': 'Bruce B. Doris',
    'NumberofPatentFamilies': 522,
    'Country': 'USA',
    'Number of INPADOC patents': 867,
    'Active': '1995-2016',
    'Mainfieldsofinvention': 'Integrated Circuits, CMOS, DRAM,
Semiconductor device fabrication',
  },
  {
    'Inventor': 'Clyde C. Farmer',
    'NumberofPatentFamilies': 513,
    'Country': 'USA',
    'Number of INPADOC patents': 830,
    'Active': '18??(b)-19??(d)',
    'Mainfieldsofinvention': 'Railway air brakes',
  },
  {

```



```

    'Inventor': 'Heinz Focke',
    'NumberofPatentFamilies': 512,
    'Country': 'Germany',
    'Number of INPADOC patents': 2896,
    'Active': '1976-2013',
    'Mainfieldsofinvention': 'Cigarette packaging',
  },
  {
    'Inventor': 'Mark I. Gardner',
    'NumberofPatentFamilies': 511,
    'Country': 'USA',
    'Number of INPADOC patents': 587,
    'Active': '1994-2010',
    'Mainfieldsofinvention': 'Consumer electronics, Energy, Computers, Semiconductors, Physics',
  },
  {
    'Inventor': 'Ravi K. Arimilli',
    'NumberofPatentFamilies': 506,
    'Country': 'India',
    'Number of INPADOC patents': 767,
    'Active': '1992-2016',
    'Mainfieldsofinvention': 'Computer architecture, Semiconductor memory, Cache coherence, Symmetric multiprocessing',
  },
  {
    'Inventor': 'Louis H. Morin',
    'NumberofPatentFamilies': 503,
    'Country': 'USA',
    'Number of INPADOC patents': 720,
    'Active': '18??(b)-19??(d)',
    'Mainfieldsofinvention': 'Fasteners, Locks, Bobbins',
  },
  {
    'Inventor': 'Tobin A. King',
    'NumberofPatentFamilies': 497,
    'Country': 'Australia',
    'Number of INPADOC patents': 1218,
    'Active': '2000-2015',
    'Mainfieldsofinvention': 'Printing, Digital paper, Mechanical',
  },
  {
    'Inventor': 'Eric C. Leuthardt',
    'NumberofPatentFamilies': 495,
    'Country': 'USA',
    'Number of INPADOC patents': 1274,
    'Active': '2006-2016',
    'Mainfieldsofinvention': 'Medical devices',
  },
  {
    'Inventor': 'Ali Khakifirooz',
    'NumberofPatentFamilies': 489,
    'Country': 'USA',
    'Number of INPADOC patents': 737,
    'Active': '2011-2016',
    'Mainfieldsofinvention': 'Integrated Circuits, CMOS, Semiconductor device fabrication',
  },

```

```

    },
    {
      'Inventor': 'Jack A. Mandelman',
      'NumberofPatentFamilies': 481,
      'Country': 'USA',
      'Number of INPADOC patents': 889,
      'Active': '1987-2014',
      'Mainfieldsofinvention': 'Various',
    },
    {
      'Inventor': 'Jeffrey P. Gambino',
      'NumberofPatentFamilies': 479,
      'Country': 'USA',
      'Number of INPADOC patents': 798,
      'Active': '1992-2016',
      'Mainfieldsofinvention': 'MEMS, CMOS, BiCMOS, DRAM, Image Sensors,
RF, Biosensors, 3D Integrated Circuits',
    },
    {
      'Inventor': 'John M. Santosuosso',
      'NumberofPatentFamilies': 473,
      'Country': 'USA',
      'Number of INPADOC patents': 683,
      'Active': '2001-2016',
      'Mainfieldsofinvention': 'Various',
    },
    {
      'Inventor': 'James M. Hart',
      'NumberofPatentFamilies': 464,
      'Country': 'USA',
      'Number of INPADOC patents': 1145,
      'Active': '1988-2016',
      'Mainfieldsofinvention': 'Motor vehicle transmission',
    },
    {
      'Inventor': 'Eberhard Ammermann',
      'NumberofPatentFamilies': 451,
      'Country': 'Germany',
      'Number of INPADOC patents': 5178,
      'Active': '1979-2015',
      'Mainfieldsofinvention': 'Fungicides',
    },
    {
      'Inventor': 'Thomas E. Murray',
      'NumberofPatentFamilies': 449,
      'Country': 'USA',
      'Number of INPADOC patents': 462,
      'Active': '1860(b)-1929(d)',
      'Mainfieldsofinvention': 'Electrical, HVAC, Wheels, Metal working,
Light dimmer',
    },
    {
      'Inventor': 'Akira Nakazawa',
      'NumberofPatentFamilies': 445,
      'Country': 'Australia',
      'Number of INPADOC patents': 1340,
      'Active': '1980-2016',
    }
  ],
  'Total': 10,
  'TotalPatentFamilies': 4490,
  'TotalINPADOCPatents': 20000,
  'TotalActive': 10,
  'TotalMainfieldsofinvention': 10
}

```

```

    'Mainfieldsofinvention': 'Printing, Mechanical',
  },
  {
    'Inventor': 'Hongyong Zhang',
    'NumberofPatentFamilies': 440,
    'Country': 'Japan',
    'Number of INPADOC patents': 858,
    'Active': '1993-2016',
    'Mainfieldsofinvention': 'Thin film transistors, Liquid crystal
displays',
  },
  {
    'Inventor': 'Ronald S. Cok',
    'NumberofPatentFamilies': 436,
    'Country': 'USA',
    'Number of INPADOC patents': 747,
    'Active': '1986-2016',
    'Mainfieldsofinvention': 'OLED displays; image processing',
  },
  {
    'Inventor': 'fe',
    'NumberofPatentFamilies': 430,
    'Country': 'USA',
    'Number of INPADOC patents': 1759,
    'Active': '1983-2016',
    'Mainfieldsofinvention': 'Biotechnology, Drug delivery, Tissue
engineering',
  },
  {
    'Inventor': 'Scott H. Wittkopp',
    'NumberofPatentFamilies': 429,
    'Country': 'USA',
    'Number of INPADOC patents': 1010,
    'Active': '2001-2016',
    'Mainfieldsofinvention': 'Motor vehicle transmission',
  },
  {
    'Inventor': 'John Hays Hammond, Jr.',
    'NumberofPatentFamilies': 417,
    'Country': 'USA',
    'Number of INPADOC patents': 460,
    'Active': '1888(b)-1965(d)',
    'Mainfieldsofinvention': 'Radio control, Radio communications,
Torpedoes',
  },
  {
    'Inventor': 'Wilhelm Brandes',
    'NumberofPatentFamilies': 411,
    'Country': 'Germany',
    'Number of INPADOC patents': 2923,
    'Active': '1976-2010',
    'Mainfieldsofinvention': 'Fungicides',
  },
  {
    'Inventor': 'Anthony K. Stamper',
    'NumberofPatentFamilies': 411,
    'Country': 'USA',

```

```

    'Number of INPADOC patents': 726,
    'Active': '1998-2016',
    'Mainfieldsofinvention': 'MEMS, CMOS, BiCMOS, Silicon-germanium',
  },
  {
    'Inventor': 'Hossein Eslambolchi',
    'NumberofPatentFamilies': 410,
    'Country': 'USA',
    'Number of INPADOC patents': 631,
    'Active': '1993-2016',
    'Mainfieldsofinvention': 'Telecommunications, Network intelligence,
information Technology, communications technology',
  },
  {
    'Inventor': 'Stanford R. Ovshinsky',
    'NumberofPatentFamilies': 400,
    'Country': 'USA',
    'Number of INPADOC patents': 1649,
    'Active': '1922 (b) -2012 (d) ',
    'Mainfieldsofinvention': 'Batteries, Solar cells, Liquid crystal
displays, Hydrogen fuel cells, Computer data storage',
  },
  {
    'Inventor': 'Victoria Y. H. Wood',
    'NumberofPatentFamilies': 400,
    'Country': 'USA',
    'Number of INPADOC patents': 1045,
    'Active': '2009-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Josef Theurer',
    'NumberofPatentFamilies': 388,
    'Country': 'Austria',
    'Number of INPADOC patents': 5085,
    'Active': '1976-2016',
    'Mainfieldsofinvention': 'Railroad maintenance machines',
  },
  {
    'Inventor': 'Cary L. Bates',
    'NumberofPatentFamilies': 384,
    'Country': 'USA',
    'Number of INPADOC patents': 570,
    'Active': '1994-2016',
    'Mainfieldsofinvention': 'Programming tools, DBX, Memory debuggers',
  },
  {
    'Inventor': 'David V. Horak',
    'NumberofPatentFamilies': 380,
    'Country': 'USA',
    'Number of INPADOC patents': 616,
    'Active': '1992-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Kai-Li Jiang',
    'NumberofPatentFamilies': 379,

```

```

        'Country': 'China',
        'Number of INPADOC patents': 829,
        'Active': '2006-2016',
        'Mainfieldsofinvention': 'Carbon nanotubes and applications of
carbon nanotubes',
    },
    {
        'Inventor': 'Hans-Joachim Santel',
        'NumberofPatentFamilies': 377,
        'Country': 'Germany',
        'Number of INPADOC patents': 2623,
        'Active': '1986-2013',
        'Mainfieldsofinvention': 'Herbicides, Pesticides, Organic
chemistry',
    },
    {
        'Inventor': 'Xuemin (Sherman) Chen',
        'NumberofPatentFamilies': 377,
        'Country': 'USA',
        'Number of INPADOC patents': 1151,
        'Active': '1997-2017',
        'Mainfieldsofinvention': 'Computer networks, Integrated Circuits,
Signal Processing',
    },
    {
        'Inventor': 'George P. Liang',
        'NumberofPatentFamilies': 375,
        'Country': 'China',
        'Number of INPADOC patents': 508,
        'Active': '1983-2016',
        'Mainfieldsofinvention': 'Gas turbine cooling',
    },
    {
        'Inventor': 'Gisela Lorenz',
        'NumberofPatentFamilies': 374,
        'Country': 'Germany',
        'Number of INPADOC patents': 4155,
        'Active': '1990-2015',
        'Mainfieldsofinvention': 'Fungicides, Organic chemistry',
    },
    {
        'Inventor': 'Garry R. Jackson',
        'NumberofPatentFamilies': 367,
        'Country': 'Australia',
        'Number of INPADOC patents': 656,
        'Active': '2001-2016',
        'Mainfieldsofinvention': 'Printing, Mechanical',
    },
    {
        'Inventor': 'Paul W. Dent',
        'NumberofPatentFamilies': 362,
        'Country': 'USA',
        'Number of INPADOC patents': 2252,
        'Active': '1984-2015',
        'Mainfieldsofinvention': 'Wireless communications',
    },
    {

```

```

    'Inventor': 'George Westinghouse',
    'NumberofPatentFamilies': 361,
    'Country': 'USA',
    'Number of INPADOC patents': 'NA',
    'Active': '1846(b)-1914(d)',
    'Mainfieldsofinvention': 'Electric power, Electricity meter, Railway
air brake, Steam engines',
  },
  {
    'Inventor': 'Wael W. Diab',
    'NumberofPatentFamilies': 358,
    'Country': 'USA',
    'Number of INPADOC patents': 774,
    'Active': '2003-2016',
    'Mainfieldsofinvention': 'Computer networks',
  },
  {
    'Inventor': 'Devendra K. Sadana',
    'NumberofPatentFamilies': 356,
    'Country': 'India',
    'Number of INPADOC patents': 794,
    'Active': '1983-2016',
    'Mainfieldsofinvention': 'Solar cells, OLED, Integrated Circuits,
CMOS, DRAM, LEDs',
  },
  {
    'Inventor': 'Vincent J. Zimmer',
    'NumberofPatentFamilies': 354,
    'Country': 'USA',
    'Number of INPADOC patents': 972,
    'Active': '1999-2016',
    'Mainfieldsofinvention': 'Computer software and firmware',
  },
  {
    'Inventor': 'Robert R. Schmidt',
    'NumberofPatentFamilies': 350,
    'Country': 'Germany',
    'Number of INPADOC patents': 2467,
    'Active': '1971-2005',
    'Mainfieldsofinvention': 'Herbicides, Fungicides, Organic
chemistry',
  },
  {
    'Inventor': 'Norman M. Berry',
    'NumberofPatentFamilies': 347,
    'Country': 'Australia',
    'Number of INPADOC patents': 516,
    'Active': '2006-2016',
    'Mainfieldsofinvention': 'Printing, Mechanical',
  },
  {
    'Inventor': 'Chih-Chao Yang',
    'NumberofPatentFamilies': 345,
    'Country': 'USA',
    'Number of INPADOC patents': 690,
    'Active': '2003-2016',
    'Mainfieldsofinvention': 'Integrated Circuits',
  },

```

```

    },
    {
      'Inventor': 'Gregory J. Boss',
      'NumberofPatentFamilies': 345,
      'Country': 'USA',
      'Number of INPADOC patents': 588,
      'Active': '2008-2016',
      'Mainfieldsofinvention': 'Various'
    },
    {
      'Inventor': 'Mark W. Kroll',
      'NumberofPatentFamilies': 343,
      'Country': 'USA',
      'Number of INPADOC patents': 460,
      'Active': '1987-2016',
      'Mainfieldsofinvention': 'Implantable medical devices',
    },
    {
      'Inventor': 'Brian M. O\'Connell',
      'NumberofPatentFamilies': 331,
      'Country': 'USA',
      'Number of INPADOC patents': 592,
      'Active': '2009-2016',
      'Mainfieldsofinvention': 'Various',
    },
    {
      'Inventor': 'William Daniel Hillis',
      'NumberofPatentFamilies': 328,
      'Country': 'USA',
      'Number of INPADOC patents': 229,
      'Active': '1986-2016',
      'Mainfieldsofinvention': 'Various',
    },
    {
      'Inventor': 'Brent A. Anderson',
      'NumberofPatentFamilies': 323,
      'Country': 'USA',
      'Number of INPADOC patents': 454,
      'Active': '2001-2016',
      'Mainfieldsofinvention': 'Semiconductor device fabrication,
Semiconductor memory, Semiconductor device',
    },
    {
      'Inventor': 'Jeffrey E. Stahmann',
      'NumberofPatentFamilies': 321,
      'Country': 'USA',
      'Number of INPADOC patents': 640,
      'Active': '1994-2016',
      'Mainfieldsofinvention': 'Medical devices',
    },
    {
      'Inventor': 'Carl J. Radens',
      'NumberofPatentFamilies': 317,
      'Country': 'USA',
      'Number of INPADOC patents': 636,
      'Active': '1994-2016',
    }
  ]

```

```

    'Mainfieldsofinvention': 'Integrated Circuits, CMOS, DRAM,
Semiconductor device fabrication',
  },
  {
    'Inventor': 'Clifford A. Pickover',
    'NumberofPatentFamilies': 317,
    'Country': 'USA',
    'Number of INPADOC patents': 653,
    'Active': '1992-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Liang Liu',
    'NumberofPatentFamilies': 310,
    'Country': 'China',
    'Number of INPADOC patents': 777,
    'Active': '2005-2016',
    'Mainfieldsofinvention': 'Carbon nanotubes and applications of
carbon nanotubes',
  },
  {
    'Inventor': 'Steven L. Teig',
    'NumberofPatentFamilies': 307,
    'Country': 'USA',
    'Number of INPADOC patents': 366,
    'Active': '1995-2016',
    'Mainfieldsofinvention': 'Integrated Circuits',
  },
  {
    'Inventor': 'Victoria Smith',
    'NumberofPatentFamilies': 305,
    'Country': 'USA',
    'Number of INPADOC patents': 2040,
    'Active': '2006-2016',
    'Mainfieldsofinvention': 'Proteins, Antibodies',
  },
  {
    'Inventor': 'Robert G. LeTourneau',
    'NumberofPatentFamilies': 299,
    'Country': 'USA',
    'Number of INPADOC patents': 'NA',
    'Active': '1888(b)-1969(d)',
    'Mainfieldsofinvention': 'Earthworks (engineering), Heavy Equipment,
Industrial Machinery',
  },
  {
    'Inventor': 'William R. Tonti',
    'NumberofPatentFamilies': 291,
    'Country': 'USA',
    'Number of INPADOC patents': 441,
    'Active': '1994-2016',
    'Mainfieldsofinvention': 'Integrated Circuits, CMOS, DRAM,
Semiconductor device fabrication',
  },
  {
    'Inventor': 'Keith R. Walker',
    'NumberofPatentFamilies': 282,

```



```

    'Country': 'Saudi Arabia',
    'Number of INPADOC patents': 318,
    'Active': '2003-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Victor S. Moore',
    'NumberofPatentFamilies': 280,
    'Country': 'USA',
    'Number of INPADOC patents': 428,
    'Active': '1982-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Hanson S. Gifford III',
    'NumberofPatentFamilies': 276,
    'Country': 'USA',
    'Number of INPADOC patents': 795,
    'Active': '1987-2016',
    'Mainfieldsofinvention': 'Medical Devices',
  },
  {
    'Inventor': 'Daniel J. Winarski',
    'NumberofPatentFamilies': 275,
    'Country': 'USA',
    'Number of INPADOC patents': 506,
    'Active': '1982-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Adam Heller',
    'NumberofPatentFamilies': 272,
    'Country': 'Romania',
    'Number of INPADOC patents': 711,
    'Active': '1968-2016',
    'Mainfieldsofinvention': 'Solar cells, Glucose meters, Lasers',
  },
  {
    'Inventor': 'Lisa Seacat DeLuca',
    'NumberofPatentFamilies': 271,
    'Country': 'USA',
    'Number of INPADOC patents': 385,
    'Active': '2009-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Brent Keeth',
    'NumberofPatentFamilies': 270,
    'Country': 'USA',
    'Number of INPADOC patents': 470,
    'Active': '1994-2016',
    'Mainfieldsofinvention': 'Integrated Circuits, CMOS, DRAM',
  },
  {
    'Inventor': 'Hartley Owen',
    'NumberofPatentFamilies': 267,
    'Country': 'USA',

```

```

    'Number of INPADOC patents': 751,
    'Active': '1976-2010',
    'Mainfieldsofinvention': 'Fluid catalytic cracking',
  },
  {
    'Inventor': 'Michael A. Rothman',
    'NumberofPatentFamilies': 256,
    'Country': 'USA',
    'Number of INPADOC patents': 687,
    'Active': '2001-2017',
    'Mainfieldsofinvention': 'Computer software and firmware',
  },
  {
    'Inventor': 'Yoshihiro Kikuchi',
    'NumberofPatentFamilies': 255,
    'Country': 'Japan',
    'Number of INPADOC patents': 1120,
    'Active': '1994-2015',
    'Mainfieldsofinvention': 'Video processing',
  },
  {
    'Inventor': 'Kulvir S. Bhogal',
    'NumberofPatentFamilies': 252,
    'Country': 'USA',
    'Number of INPADOC patents': 486,
    'Active': '2003-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Bengt Lindoff',
    'NumberofPatentFamilies': 248,
    'Country': 'Sweden',
    'Number of INPADOC patents': 1658,
    'Active': '2000-2017',
    'Mainfieldsofinvention': 'Wireless communications',
  },
  {
    'Inventor': 'Nobuyuki Taniguchi',
    'NumberofPatentFamilies': 245,
    'Country': 'Japan',
    'Number of INPADOC patents': 967,
    'Active': '1979-2015',
    'Mainfieldsofinvention': 'Cameras',
  },
  {
    'Inventor': 'Dean L. Kamen',
    'NumberofPatentFamilies': 243,
    'Country': 'USA',
    'Number of INPADOC patents': 1186,
    'Active': '1979-2016',
    'Mainfieldsofinvention': 'Battery-powered electric vehicles, Medical
    devices, Stirling engines, Water purification, Wheelchairs',
  },
  {
    'Inventor': 'Philip S. Yu',
    'NumberofPatentFamilies': 236,
    'Country': 'USA',

```

```

    'Number of INPADOC patents': 158,
    'Active': '1982-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Rajiv V. Joshi',
    'NumberofPatentFamilies': 235,
    'Country': 'USA',
    'Number of INPADOC patents': 354,
    'Active': '1986-2016',
    'Mainfieldsofinvention': 'Electronics, analytics',
  },
  {
    'Inventor': 'Lawrence A. Clevenger',
    'NumberofPatentFamilies': 235,
    'Country': 'USA',
    'Number of INPADOC patents': 526,
    'Active': '1996-2017',
    'Mainfieldsofinvention': 'Semiconductor, Cognitive, Memory,
Security, Analytics',
  },
  {
    'Inventor': 'Johnny M. Shieh',
    'NumberofPatentFamilies': 231,
    'Country': 'USA',
    'Number of INPADOC patents': 444,
    'Active': '1996-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Takeshi Chujoh',
    'NumberofPatentFamilies': 229,
    'Country': 'Japan',
    'Number of INPADOC patents': 1065,
    'Active': '1995-2016',
    'Mainfieldsofinvention': 'Video processing',
  },
  {
    'Inventor': 'Bran Ferren',
    'NumberofPatentFamilies': 225,
    'Country': 'USA',
    'Number of INPADOC patents': 589,
    'Active': '1986-2017',
    'Mainfieldsofinvention': 'Computers, Consumer Electronics, Optical
Systems, Medical, User Interfaces, Automotive',
  },
  {
    'Inventor': 'Paul Ian Mackey',
    'NumberofPatentFamilies': 220,
    'Country': 'Australia',
    'Number of INPADOC patents': 246,
    'Active': '2008-2016',
    'Mainfieldsofinvention': 'Printing, Mechanical',
  },
  {
    'Inventor': 'Louis Rosenberg',
    'NumberofPatentFamilies': 218,

```

```

    'Country': 'USA',
    'Number of INPADOC patents': 444,
    'Active': '1995-2016',
    'Mainfieldsofinvention': 'Augmented Reality, Virtual Reality, A.I.,
HCI',
  },
  {
    'Inventor': 'Thomas J. Kennedy III',
    'NumberofPatentFamilies': 218,
    'Country': 'USA',
    'Number of INPADOC patents': 513,
    'Active': '1992-2016',
    'Mainfieldsofinvention': 'Sporting Goods, Wind Turbines',
  },
  {
    'Inventor': 'Gerald F. McBrearty',
    'NumberofPatentFamilies': 213,
    'Country': 'USA',
    'Number of INPADOC patents': 387,
    'Active': '1997-2016',
    'Mainfieldsofinvention': 'Various',
  },
  {
    'Inventor': 'Esmael H. Dinan',
    'NumberofPatentFamilies': 208,
    'Country': 'USA',
    'Number of INPADOC patents': 344,
    'Active': '2000-2017',
    'Mainfieldsofinvention': 'Communication Networks',
  },
  {
    'Inventor': 'Imad Libbus',
    'NumberofPatentFamilies': 207,
    'Country': 'USA',
    'Number of INPADOC patents': 472,
    'Active': '2007-2017',
    'Mainfieldsofinvention': 'Medical devices',
  },
  {
    'Inventor': 'Hiroshi (You) Yoshioka',
    'NumberofPatentFamilies': 205,
    'Country': 'Japan',
    'Number of INPADOC patents': 181,
    'Active': '1997-2015',
    'Mainfieldsofinvention': 'Cameras',
  },
  {
    'Inventor': 'Patrick B. Usoro',
    'NumberofPatentFamilies': 205,
    'Country': 'USA',
    'Number of INPADOC patents': 343,
    'Active': '1999-2016',
    'Mainfieldsofinvention': 'Transmissions, Hybrid Powertrains, Vehicle
Thermal Management',
  },
  {
    'Inventor': 'Gregory McAvoy',

```

```

        'NumberofPatentFamilies': 205,
        'Country': 'Australia',
        'Number of INPADOC patents': 433,
        'Active': '2003-2014',
        'Mainfieldsofinvention': 'Printing, MEMS',
    },
    {
        'Inventor': 'Sebastian T Ventrone',
        'NumberofPatentFamilies': 204,
        'Country': 'USA',
        'Number of INPADOC patents': 283,
        'Active': '1989-2017',
        'Mainfieldsofinvention': 'Semiconductor, Logic, Architecture',
    },
    {
        'Inventor': 'Dorin Comaniciu',
        'NumberofPatentFamilies': 200,
        'Country': 'USA',
        'Number of INPADOC patents': 452,
        'Active': '2003-2017',
        'Mainfieldsofinvention': 'Machine Intelligence, Medical Imaging,
Image-Guided Surgery, Computer Vision',
    }
];
export interface ColumnSpanDataType {
    EmployeeID: number;
    EmployeeName: string;
    '9:00': string;
    '9:30': string;
    '10:00': string;
    '10:30': string;
    '11:00': string;
    '11:30': string;
    '12:00': string;
    '12:30': string;
    '1:00': string;
    '1:30': string;
    '2:00': string;
    '2:30': string;
    '3:00': string;
    '3:30': string;
    '4:00': string;
    '4:30': string;
    '5:00': string;
}
export let columnSpanData: ColumnSpanDataType[] = [
    {
        EmployeeID: 10001,
        EmployeeName: 'Davolio',
        '9:00': 'Analysis Tasks',
        '9:30': 'Analysis Tasks',
        '10:00': 'Team Meeting',
        '10:30': 'Testing',
        '11:00': 'Development',
        '11:30': 'Development',
        '12:00': 'Development',
        '12:30': 'Support',
    }
];

```

```

        '1:00': 'Lunch Break',
        '1:30': 'Lunch Break',
        '2:00': 'Lunch Break',
        '2:30': 'Testing',
        '3:00': 'Testing',
        '3:30': 'Development',
        '4:00': 'Conference',
        '4:30': 'Team Meeting',
        '5:00': 'Team Meeting'
    },
    {
        EmployeeID: 10002,
        EmployeeName: 'Buchanan',
        '9:00': 'Task Assign',
        '9:30': 'Support',
        '10:00': 'Support',
        '10:30': 'Support',
        '11:00': 'Testing',
        '11:30': 'Testing',
        '12:00': 'Testing',
        '12:30': 'Testing',
        '1:00': 'Lunch Break',
        '1:30': 'Lunch Break',
        '2:00': 'Lunch Break',
        '2:30': 'Development',
        '3:00': 'Development',
        '3:30': 'Check Mail',
        '4:00': 'Check Mail',
        '4:30': 'Team Meeting',
        '5:00': 'Team Meeting'
    },
    {
        EmployeeID: 10003,
        EmployeeName: 'Fuller',
        '9:00': 'Check Mail',
        '9:30': 'Check Mail',
        '10:00': 'Check Mail',
        '10:30': 'Analysis Tasks',
        '11:00': 'Analysis Tasks',
        '11:30': 'Support',
        '12:00': 'Support',
        '12:30': 'Support',
        '1:00': 'Lunch Break',
        '1:30': 'Lunch Break',
        '2:00': 'Lunch Break',
        '2:30': 'Development',
        '3:00': 'Development',
        '3:30': 'Team Meeting',
        '4:00': 'Team Meeting',
        '4:30': 'Development',
        '5:00': 'Development'
    },
    {
        EmployeeID: 10004,
        EmployeeName: 'Leverling',
        '9:00': 'Testing',
        '9:30': 'Check Mail',

```

```

        '10:00': 'Check Mail',
        '10:30': 'Support',
        '11:00': 'Testing',
        '11:30': 'Testing',
        '12:00': 'Testing',
        '12:30': 'Testing',
        '1:00': 'Lunch Break',
        '1:30': 'Lunch Break',
        '2:00': 'Lunch Break',
        '2:30': 'Development',
        '3:00': 'Development',
        '3:30': 'Check Mail',
        '4:00': 'Conference',
        '4:30': 'Conference',
        '5:00': 'Team Meeting'
    },
    {
        EmployeeID: 10005,
        EmployeeName: 'Peacock',
        '9:00': 'Task Assign',
        '9:30': 'Task Assign',
        '10:00': 'Task Assign',
        '10:30': 'Task Assign',
        '11:00': 'Check Mail',
        '11:30': 'Support',
        '12:00': 'Support',
        '12:30': 'Support',
        '1:00': 'Lunch Break',
        '1:30': 'Lunch Break',
        '2:00': 'Lunch Break',
        '2:30': 'Development',
        '3:00': 'Development',
        '3:30': 'Team Meeting',
        '4:00': 'Team Meeting',
        '4:30': 'Testing',
        '5:00': 'Testing'
    },
    {
        EmployeeID: 10006,
        EmployeeName: 'Janet',
        '9:00': 'Testing',
        '9:30': 'Testing',
        '10:00': 'Support',
        '10:30': 'Support',
        '11:00': 'Support',
        '11:30': 'Team Meeting',
        '12:00': 'Team Meeting',
        '12:30': 'Team Meeting',
        '1:00': 'Lunch Break',
        '1:30': 'Lunch Break',
        '2:00': 'Lunch Break',
        '2:30': 'Development',
        '3:00': 'Development',
        '3:30': 'Team Meeting',
        '4:00': 'Team Meeting',
        '4:30': 'Development',
        '5:00': 'Development'
    }

```

```

    },
    {
        EmployeeID: 10007,
        EmployeeName: 'Suyama',
        '9:00': 'Analysis Tasks',
        '9:30': 'Analysis Tasks',
        '10:00': 'Testing',
        '10:30': 'Development',
        '11:00': 'Development',
        '11:30': 'Testing',
        '12:00': 'Testing',
        '12:30': 'Testing',
        '1:00': 'Lunch Break',
        '1:30': 'Lunch Break',
        '2:00': 'Lunch Break',
        '2:30': 'Support',
        '3:00': 'Build',
        '3:30': 'Build',
        '4:00': 'Check Mail',
        '4:30': 'Check Mail',
        '5:00': 'Check Mail'
    },
    {
        EmployeeID: 10008,
        EmployeeName: 'Robert',
        '9:00': 'Task Assign',
        '9:30': 'Task Assign',
        '10:00': 'Task Assign',
        '10:30': 'Development',
        '11:00': 'Development',
        '11:30': 'Development',
        '12:00': 'Testing',
        '12:30': 'Support',
        '1:00': 'Lunch Break',
        '1:30': 'Lunch Break',
        '2:00': 'Lunch Break',
        '2:30': 'Check Mail',
        '3:00': 'Check Mail',
        '3:30': 'Check Mail',
        '4:00': 'Team Meeting',
        '4:30': 'Team Meeting',
        '5:00': 'Build'
    },
    {
        EmployeeID: 10009,
        EmployeeName: 'Andrew',
        '9:00': 'Check Mail',
        '9:30': 'Team Meeting',
        '10:00': 'Team Meeting',
        '10:30': 'Support',
        '11:00': 'Testing',
        '11:30': 'Development',
        '12:00': 'Development',
        '12:30': 'Development',
        '1:00': 'Lunch Break',
        '1:30': 'Lunch Break',
        '2:00': 'Lunch Break',
    }

```



```

        '2:30': 'Check Mail',
        '3:00': 'Check Mail',
        '3:30': 'Check Mail',
        '4:00': 'Team Meeting',
        '4:30': 'Development',
        '5:00': 'Development'
    },
    {
        EmployeeID: 10010,
        EmployeeName: 'Michael',
        '9:00': 'Task Assign',
        '9:30': 'Task Assign',
        '10:00': 'Task Assign',
        '10:30': 'Analysis Tasks',
        '11:00': 'Analysis Tasks',
        '11:30': 'Development',
        '12:00': 'Development',
        '12:30': 'Development',
        '1:00': 'Lunch Break',
        '1:30': 'Lunch Break',
        '2:00': 'Lunch Break',
        '2:30': 'Testing',
        '3:00': 'Testing',
        '3:30': 'Testing',
        '4:00': 'Build',
        '4:30': 'Build',
        '5:00': 'Build'
    }
];

```

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
ColumnSeries, DateTime } from '@syncfusion/ej2-react-charts';
import { ColumnDirective, ColumnsDirective, GridComponent, Page, Selection }
from '@syncfusion/ej2-react-grids';
function App() {
    const pageSettings = { pageSize: 10 };
    var grid;
    var chart;
    const primaryxAxis = { valueType: 'DateTime', intervalType: 'Days' };
    function actionComplete(args) {
        if (args.requestType === 'paging') {
            chart.series[0].dataSource = grid.getCurrentViewRecords();
            chart.refresh();
        }
    };
    function dataBound(args) {
        chart.series[0].dataSource = grid.getCurrentViewRecords();
    };
    const data = [
        {
            'OrderDate': '1996-07-06',

```

```

        'ShippedDate': '1996-07-16',
        'Freight': 32.38
    },
    {
        'OrderDate': '1996-07-07',
        'ShippedDate': '1996-07-10',
        'Freight': 11.61
    },
    {
        'OrderDate': '1996-07-08',
        'ShippedDate': '1996-07-12',
        'Freight': 65.83
    },
    {
        'OrderDate': '1996-07-09',
        'ShippedDate': '1996-07-15',
        'Freight': 41.34
    },
    {
        'OrderDate': '1996-07-10',
        'ShippedDate': '1996-07-11',
        'Freight': 51.3
    },
    {
        'OrderDate': '1996-07-11',
        'ShippedDate': '1996-07-16',
        'Freight': 58.17
    },
    {
        'OrderDate': '1996-07-12',
        'ShippedDate': '1996-07-23',
        'Freight': 22.98
    },
    {
        'OrderDate': '1996-07-13',
        'ShippedDate': '1996-07-15',
        'Freight': 148.33
    },
    {
        'OrderDate': '1996-07-14',
        'ShippedDate': '1996-07-17',
        'Freight': 13.97
    },
    {
        'OrderDate': '1996-07-15',
        'ShippedDate': '1996-07-22',
        'Freight': 81.91
    },
    {
        'OrderDate': '1996-07-16',
        'ShippedDate': '1996-07-23',
        'Freight': 140.51
    },
    {
        'OrderDate': '1996-07-17',
        'ShippedDate': '1996-07-25',
        'Freight': 3.25
    }

```

```
},
{
  'OrderDate': '1996-07-18',
  'ShippedDate': '1996-07-29',
  'Freight': 55.09
},
{
  'OrderDate': '1996-07-19',
  'ShippedDate': '1996-07-30',
  'Freight': 3.05
},
{
  'OrderDate': '1996-07-20',
  'ShippedDate': '1996-07-25',
  'Freight': 48.29
},
{
  'OrderDate': '1996-07-21',
  'ShippedDate': '1996-07-31',
  'Freight': 146.06
},
{
  'OrderDate': '1996-07-22',
  'ShippedDate': '1996-08-23',
  'Freight': 3.67
},
{
  'OrderDate': '1996-07-23',
  'ShippedDate': '1996-08-12',
  'Freight': 55.28
},
{
  'OrderDate': '1996-07-24',
  'ShippedDate': '1996-07-31',
  'Freight': 25.73
},
{
  'OrderDate': '1996-07-25',
  'ShippedDate': '1996-08-06',
  'Freight': 208.58
},
{
  'OrderDate': '1996-07-26',
  'ShippedDate': '1996-08-02',
  'Freight': 66.29
},
{
  'OrderDate': '1996-07-27',
  'ShippedDate': '1996-08-09',
  'Freight': 4.56
},
{
  'OrderDate': '1996-07-29',
  'ShippedDate': '1996-08-02',
  'Freight': 136.54
},
{
```

```
'OrderDate': '1996-07-30',
'ShippedDate': '1996-08-30',
'Freight': 4.54
},
{
  'OrderDate': '1996-07-31',
  'ShippedDate': '1996-08-06',
  'Freight': 98.03
},
{
  'OrderDate': '1996-08-01',
  'ShippedDate': '1996-08-12',
  'Freight': 76.07
},
{
  'OrderDate': '1996-08-02',
  'ShippedDate': '1996-08-16',
  'Freight': 6.01
},
{
  'OrderDate': '1996-08-03',
  'ShippedDate': '1996-08-09',
  'Freight': 26.93
},
{
  'OrderDate': '1996-08-04',
  'ShippedDate': '1996-08-14',
  'Freight': 13.84
},
{
  'OrderDate': '1996-08-05',
  'ShippedDate': '1996-08-13',
  'Freight': 125.77
},
{
  'OrderDate': '1996-08-06',
  'ShippedDate': '1996-08-16',
  'Freight': 92.69
},
{
  'OrderDate': '1996-08-07',
  'ShippedDate': '1996-08-16',
  'Freight': 25.83
},
{
  'OrderDate': '1996-08-08',
  'ShippedDate': '1996-09-12',
  'Freight': 8.98
},
{
  'OrderDate': '1996-08-09',
  'ShippedDate': '1996-08-21',
  'Freight': 2.94
},
{
  'OrderDate': '1996-08-10',
  'ShippedDate': '1996-08-21',
```

```

        'Freight': 12.69
    },
    {
        'OrderDate': '1996-08-11',
        'ShippedDate': '1996-08-28',
        'Freight': 12.76
    },
    {
        'OrderDate': '1996-08-12',
        'ShippedDate': '1996-09-03',
        'Freight': 7.45
    },
    {
        'OrderDate': '1996-08-13',
        'ShippedDate': '1996-08-28',
        'Freight': 22.77
    },
    {
        'OrderDate': '1996-08-14',
        'ShippedDate': '1996-09-03',
        'Freight': 79.7
    },
    {
        'OrderDate': '1996-08-15',
        'ShippedDate': '1996-09-04',
        'Freight': 6.4
    }
];
return (<div className='row'>
    <div className="col-sm-4">
        <GridComponent ref={g => grid = g} id='grid' dataSource={data}
allowPaging={true} pageSettings={pageSettings}
actionComplete={actionComplete.bind(this)} dataBound={dataBound.bind(this)}>
            <ColumnsDirective>
                <ColumnDirective field='OrderDate' headerText='Order
Date' width='130' format='yMd' textAlign="Right" />
                <ColumnDirective field='Freight' width='120' format='C2'
textAlign='Right' />
            </ColumnsDirective>
            <Inject services={[Page, Selection]} />
        </GridComponent>
    </div>
    <div className="col-sm-4">
        <ChartComponent ref={g => chart = g} id='charts'
primaryXAxis={primaryxAxis}>
            <Inject services={[ColumnSeries, DateTime]} />
            <SeriesCollectionDirective>
                <SeriesDirective name='Germany' xName='OrderDate'
yName='Freight' width={2} type='Column' marker={{ visible: true, width: 10,
height: 10 }}>
            </SeriesDirective>
        </SeriesCollectionDirective>
    </ChartComponent>
    </div>
</div>);
}
;

```

```
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
AxisModel, Chart, ColumnSeries, DateTime } from '@syncfusion/ej2-react-
charts';
import { ColumnDirective, ColumnsDirective, GridComponent, Page,
PageSettingsModel, Selection, Grid, ActionEventArgs } from '@syncfusion/ej2-
react-grids';
function App() {
    const pageSettings: PageSettingsModel = { pageSize: 10 }
    var grid: Grid | null;
    var chart: Chart | null;
    const primaryXAxis: AxisModel = { valueType: 'DateTime', intervalType:
'Days' };
    function actionComplete(args: ActionEventArgs) {
        if (args.requestType === 'paging') {
            (chart as Chart).series[0].dataSource = (grid as
Grid).getCurrentViewRecords();
            (chart as Chart).refresh();
        }
    };
    function dataBound(args: any): void {
        (chart as Chart).series[0].dataSource = (grid as
Grid).getCurrentViewRecords();
    };
    const data: any[] = [
        {
            'OrderDate': '1996-07-06',
            'ShippedDate': '1996-07-16',
            'Freight': 32.38
        },
        {
            'OrderDate': '1996-07-07',
            'ShippedDate': '1996-07-10',
            'Freight': 11.61
        },
        {
            'OrderDate': '1996-07-08',
            'ShippedDate': '1996-07-12',
            'Freight': 65.83
        },
        {
            'OrderDate': '1996-07-09',
            'ShippedDate': '1996-07-15',
            'Freight': 41.34
        },
        {
            'OrderDate': '1996-07-10',
            'ShippedDate': '1996-07-11',

```

```
        'Freight': 51.3
    },
    {
        'OrderDate': '1996-07-11',
        'ShippedDate': '1996-07-16',
        'Freight': 58.17
    },
    {
        'OrderDate': '1996-07-12',
        'ShippedDate': '1996-07-23',
        'Freight': 22.98
    },
    {
        'OrderDate': '1996-07-13',
        'ShippedDate': '1996-07-15',
        'Freight': 148.33
    },
    {
        'OrderDate': '1996-07-14',
        'ShippedDate': '1996-07-17',
        'Freight': 13.97
    },
    {
        'OrderDate': '1996-07-15',
        'ShippedDate': '1996-07-22',
        'Freight': 81.91
    },
    {
        'OrderDate': '1996-07-16',
        'ShippedDate': '1996-07-23',
        'Freight': 140.51
    },
    {
        'OrderDate': '1996-07-17',
        'ShippedDate': '1996-07-25',
        'Freight': 3.25
    },
    {
        'OrderDate': '1996-07-18',
        'ShippedDate': '1996-07-29',
        'Freight': 55.09
    },
    {
        'OrderDate': '1996-07-19',
        'ShippedDate': '1996-07-30',
        'Freight': 3.05
    },
    {
        'OrderDate': '1996-07-20',
        'ShippedDate': '1996-07-25',
        'Freight': 48.29
    },
    {
        'OrderDate': '1996-07-21',
        'ShippedDate': '1996-07-31',
        'Freight': 146.06
    },
    ],
```

```

{
    'OrderDate': '1996-07-22',
    'ShippedDate': '1996-08-23',
    'Freight': 3.67
},
{
    'OrderDate': '1996-07-23',
    'ShippedDate': '1996-08-12',
    'Freight': 55.28
},
{
    'OrderDate': '1996-07-24',
    'ShippedDate': '1996-07-31',
    'Freight': 25.73
},
{
    'OrderDate': '1996-07-25',
    'ShippedDate': '1996-08-06',
    'Freight': 208.58
},
{
    'OrderDate': '1996-07-26',
    'ShippedDate': '1996-08-02',
    'Freight': 66.29
},
{
    'OrderDate': '1996-07-27',
    'ShippedDate': '1996-08-09',
    'Freight': 4.56
},
{
    'OrderDate': '1996-07-29',
    'ShippedDate': '1996-08-02',
    'Freight': 136.54
},
{
    'OrderDate': '1996-07-30',
    'ShippedDate': '1996-08-30',
    'Freight': 4.54
},
{
    'OrderDate': '1996-07-31',
    'ShippedDate': '1996-08-06',
    'Freight': 98.03
},
{
    'OrderDate': '1996-08-01',
    'ShippedDate': '1996-08-12',
    'Freight': 76.07
},
{
    'OrderDate': '1996-08-02',
    'ShippedDate': '1996-08-16',
    'Freight': 6.01
},
{
    'OrderDate': '1996-08-03',

```



```

        'ShippedDate': '1996-08-09',
        'Freight': 26.93
    },
    {
        'OrderDate': '1996-08-04',
        'ShippedDate': '1996-08-14',
        'Freight': 13.84
    },
    {
        'OrderDate': '1996-08-05',
        'ShippedDate': '1996-08-13',
        'Freight': 125.77
    },
    {
        'OrderDate': '1996-08-06',
        'ShippedDate': '1996-08-16',
        'Freight': 92.69
    },
    {
        'OrderDate': '1996-08-07',
        'ShippedDate': '1996-08-16',
        'Freight': 25.83
    },
    {
        'OrderDate': '1996-08-08',
        'ShippedDate': '1996-09-12',
        'Freight': 8.98
    },
    {
        'OrderDate': '1996-08-09',
        'ShippedDate': '1996-08-21',
        'Freight': 2.94
    },
    {
        'OrderDate': '1996-08-10',
        'ShippedDate': '1996-08-21',
        'Freight': 12.69
    },
    {
        'OrderDate': '1996-08-11',
        'ShippedDate': '1996-08-28',
        'Freight': 12.76
    },
    {
        'OrderDate': '1996-08-12',
        'ShippedDate': '1996-09-03',
        'Freight': 7.45
    },
    {
        'OrderDate': '1996-08-13',
        'ShippedDate': '1996-08-28',
        'Freight': 22.77
    },
    {
        'OrderDate': '1996-08-14',
        'ShippedDate': '1996-09-03',
        'Freight': 79.7
    }

```

```

    },
    {
      'OrderDate': '1996-08-15',
      'ShippedDate': '1996-09-04',
      'Freight': 6.4
    }
  ];
  return (<div className='row'>
    <div className="col-sm-4">
      <GridComponent ref={g => grid = g} id='grid' dataSource={data}
        allowPaging={true} pageSettings={pageSettings}
        actionComplete={actionComplete.bind(this)} dataBound={dataBound.bind(this)}>
        <ColumnsDirective>
          <ColumnDirective field='OrderDate' headerText='Order
Date' width='130' format='yMd' textAlign="Right" />
          <ColumnDirective field='Freight' width='120' format='C2'
        textAlign='Right' />
        </ColumnsDirective>
        <Inject services={[Page, Selection]} />
      </GridComponent>
    </div>
    <div className="col-sm-4">
      <ChartComponent ref={g => chart = g} id='charts'
        primaryXAxis={primaryxAxis}>
        <Inject services={[ColumnSeries, DateTime]} />
        <SeriesCollectionDirective>
          <SeriesDirective name='Germany' xName='OrderDate'
        yName='Freight' width={2} type='Column' marker={{ visible: true, width: 10,
        height: 10 }}>
          </SeriesDirective>
        </SeriesCollectionDirective>
      </ChartComponent>
    </div>
  </div>)
};
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Data label template in React Chart component

You can bind text and interior information for a point from dataSource other than x and y value. To change color for the background in the datalabel template, you can use `${point.text}`.

To use point.text, you have to bind the property from dataSource to name in the datalabel options.

Follow the given steps to show the table tooltip,

Step 1:

Initialize the datalabel template div as shown in the following html page,

```
<script id="index" type="text/x-template">
```

```
<div id='templateWrap' style="background-color: ${point.text}; border-radius:
3px;"><span>${point.y}</span></div>
```

</script>

Step 2:

To show that datalabel template, set the element id to the `template` property in datalabel.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, SeriesDirective, Inject,
StripLine, Legend, Category, Tooltip, DataLabel, LineSeries } from
"@syncfusion/ej2-react-charts";
function App() {
  const data = [
    { x: 10, y: 7000, color: 'red' },
    { x: 20, y: 1000, color: 'yellow' },
    { x: 30, y: 12000, color: 'orange' },
    { x: 40, y: 14000, color: 'skyblue' },
    { x: 50, y: 11000, color: 'blue' },
    { x: 60, y: 5000, color: 'green' },
    { x: 70, y: 7300, color: 'pink' },
    { x: 80, y: 9000, color: 'white' },
    { x: 90, y: 12000, color: 'magenta' },
    { x: 100, y: 14000, color: 'purple' },
    { x: 110, y: 11000, color: 'teal' },
    { x: 120, y: 5000, color: 'gray' },
  ];
  const template = chartTemplate;
  const marker = { visible: true, dataLabel: { visible: true, name:
'color', template: template } };
  function chartTemplate(args) {
    return (<div id="templateWrap" style={{ border: '1px solid black',
backgroundColor: 'red', padding: '3px 3px 3px 3px' }}>
      <div>{args.point.x}</div>
      <div>{args.point.y}</div>
    </div>);
  }
  return <ChartComponent id='charts'>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category,
StripLine]}>
      <SeriesCollectionDirective>
        <SeriesDirective dataSource={data} xName='x' yName='y' type='Line'
marker={marker}>
          </SeriesDirective>
        </SeriesCollectionDirective>
      </ChartComponent>;
    </Inject>
  </ChartComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ChartComponent, SeriesCollectionDirective, AxesDirective,
AxisDirective, SeriesDirective, Inject, StripLine, ColumnSeries, Legend,
Category, Tooltip, DataLabel, Zoom, Crosshair, LineSeries, Selection,
StripLinesDirective, StripLineDirective } from '@syncfusion/ej2-react-charts';
function App() {
  const data: any[] = [
    { x: 10, y: 7000, color: 'red' },
    { x: 20, y: 1000, color: 'yellow' },
    { x: 30, y: 12000, color: 'orange' },
    { x: 40, y: 14000, color: 'skyblue' },
    { x: 50, y: 11000, color: 'blue' },
    { x: 60, y: 5000, color: 'green' },
    { x: 70, y: 7300, color: 'pink' },
    { x: 80, y: 9000, color: 'white' },
    { x: 90, y: 12000, color: 'magenta' },
    { x: 100, y: 14000, color: 'purple' },
    { x: 110, y: 11000, color: 'teal' },
    { x: 120, y: 5000, color: 'gray' },
  ];
  const template: any = chartTemplate;
  const marker = { visible: true, dataLabel: { visible: true, name: 'color',
template: template } };
  function chartTemplate(args: any) {
    return (
      <div id="templateWrap" style={{ border: '1px solid black',
background-color: 'red', padding: '3px 3px 3px 3px' }}>
        <div>{args.point.x}</div>
        <div>{args.point.y}</div>
      </div>);
  }
  return <ChartComponent id='charts'>
    <Inject services={[LineSeries, Legend, Tooltip, DataLabel, Category,
StripLine]} />
    <SeriesCollectionDirective>
      <SeriesDirective dataSource={data} xName='x' yName='y' type='Line'
marker={marker}>
        </SeriesDirective>
      </SeriesCollectionDirective>
    </ChartComponent>
  </>
};
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

Check box

Getting Started

This section explains how to create a simple CheckBox, and configure its available functionalities in React, using React quickstart application.

Dependencies

The following list of dependencies are required to use the CheckBox component in your application.

```
`javascript
|-- @syncfusion/ej2-react-buttons
|-- @syncfusion/ej2-react-base
|-- @syncfusion/ej2-base
|-- @syncfusion/ej2-buttons
`,`
```

Installation and Configuration

You can use [Create-react-app](#) to setup the applications. To install `create-react-app` run the following command.

```
`bash
npm install -g create-react-app
`,`
```

To set-up a React application in TypeScript environment, run the following command.

```
`bash
npx create-react-app my-app --template typescript
cd my-app
npm start
`,`
```

To set-up a React application in JavaScript environment, run the following command.

```
`bash
npx create-react-app my-app
cd my-app
npm start
`,`
```

Adding Syncfusion packages

All the available Essential JS 2 packages are published in [npmjs.com](https://www.npmjs.com) public registry.

To install CheckBox component, use the following command

```
`bash
npm install @syncfusion/ej2-react-buttons --save
`,`
```

Adding CSS Reference

Import the CheckBox component's required CSS references as follows in `src/App.css`.

```
`css
@import "../node_modules/@syncfusion/ej2-base/styles/material.css";
```

```
@import "../node_modules/@syncfusion/ej2-buttons/styles/material.css";  
`
```

Adding CheckBox component to the Application

To include the CheckBox component in your application import the `CheckBoxComponent` from `ej2-react-buttons` package in `App.tsx`.

Add the CheckBox component in application as shown in below code example.

```
`ts  
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';  
import * as React from 'react';  
import './App.css';  
// Import the CheckBox.  
// To render CheckBox.  
function App() {  
  {  
    return (<div style={{marginTop: '150px'}}>  
      <CheckBoxComponent label="Default"/>  
    </div>);  
  }  
}  
export default App;  
`
```

Run the application

Run the application in the browser using the following command:

```
`  
npm start  
`
```

The following example shows a basic CheckBox component.

APP.JSX

```
import { enableRipple } from '@syncfusion/ej2-base';  
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';  
import * as React from 'react';  
import * as ReactDOM from 'react-dom';  
enableRipple(true);  
// To render CheckBox.  
function App() {  
  return (<CheckBoxComponent label="Default"/>);  
}  
export default App;
```

```
ReactDOM.render(<App />, document.getElementById('check-box'));
```

APP.TSX

```
import { enableRipple } from '@syncfusion/ej2-base';
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
// To render CheckBox.
function App() {
  return (
    <CheckBoxComponent label="Default" />
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('check-box'));
```

States in React Check box component

The Essential JS 2 CheckBox contains 3 different states visually, they are:

- Checked
- Unchecked
- Indeterminate

Checked and Unchecked

The CheckBox [checked](#) property is used to handle the checked and unchecked state. In checked state a tick mark will be added to the visualization of CheckBox.

Indeterminate

The CheckBox indeterminate state can be set through [indeterminate](#) property. CheckBox indeterminate state masks the real value of CheckBox visually. The Checkbox cannot be changed to indeterminate state through the user interface, this state can be achieved only through the property.

APP.JSX

```
import { enableRipple } from '@syncfusion/ej2-base';
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
  return <ul>
    {/* checked state. */}
    <li><CheckBoxComponent label="Checked State" checked={true}/></li>
    {/* unchecked state. */}
    <li><CheckBoxComponent label="Unchecked State"/></li>
    {/* indeterminate state. */}
    <li><CheckBoxComponent label="Indeterminate State"
indeterminate={true}/></li>
  </ul>;
}
export default App;
```

```
ReactDOM.render(<App />, document.getElementById('check-box'));
```

APP.TSX

```
import { enableRipple } from '@syncfusion/ej2-base';
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
  return (
    <ul>
      {/* checked state. */}
      <li><CheckBoxComponent label="Checked State" checked={true}
    /></li>
      {/* unchecked state. */}
      <li><CheckBoxComponent label="Unchecked State" /></li>
      {/* indeterminate state. */}
      <li><CheckBoxComponent label="Indeterminate State"
    indeterminate={true} /></li>
    </ul>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('check-box'));
```

Label and size in React Check box component

This section explains the different sizes and labels.

Label

The CheckBox caption can be defined using the [label](#) property. This reduces manual addition of label for CheckBox. You can customize the label position before or after the CheckBox through the [labelPosition](#) property.

APP.JSX

```
import { enableRipple } from '@syncfusion/ej2-base';
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
  return <ul>
    {/* Label position - Left. */}
    <li><CheckBoxComponent label="Left Side Label"
    labelPosition="Before"/></li>
    {/* Label position - Right. */}
    <li><CheckBoxComponent label="Right Side Label"
    checked={true}/></li>
  </ul>;
}
export default App;
ReactDOM.render(<App />, document.getElementById('check-box'));
```


APP.TSX

```
import { enableRipple } from '@syncfusion/ej2-base';
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    return (
        <ul>
            {/* Label position - Left. */}
            <li><CheckBoxComponent label="Left Side Label"
labelPosition="Before" /></li>
            {/* Label position - Right. */}
            <li><CheckBoxComponent label="Right Side Label" checked={true}
/></li>
        </ul>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('check-box'));
```

Size

The different CheckBox sizes available are default and small. To reduce size of the default CheckBox to small, set the [cssClass](#) property to `e-small`.

APP.JSX

```
import { enableRipple } from '@syncfusion/ej2-base';
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    return <ul>
        {/* Small CheckBox. */}
        <li><CheckBoxComponent label="Small" cssClass="e-small"/></li>
        {/* Default CheckBox. */}
        <li><CheckBoxComponent label="Default"/></li>
    </ul>;
}
export default App;
ReactDOM.render(<App />, document.getElementById('check-box'));
```

APP.TSX

```
import { enableRipple } from '@syncfusion/ej2-base';
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    return (
        <ul>
            {/* Small CheckBox. */}
            <li><CheckBoxComponent label="Small" cssClass="e-small" /></li>
        </ul>
    );
}
```

```

        { /* Default CheckBox. */ }
        <li><CheckBoxComponent label="Default" /></li>
      </ul>
    );
  }
  export default App;
  ReactDOM.render(<App />, document.getElementById('check-box'));

```

See Also

- [CheckBox customization](#)

Style and appearance in React Check box component

To modify the CheckBox appearance, you need to override the default CSS of CheckBox component. Please find the list of CSS classes and its corresponding section in CheckBox. Also, you have an option to create your own custom theme for the controls using our [Theme Studio](#).

CSS Class | Purpose of Class

- |.e-checkbox-wrapper .e-frame|To customize the checkbox frame.
- |.e-checkbox-wrapper:hover .e-frame|To customize the checkbox frame on hover.
- |.e-checkbox-wrapper .e-label|To customize the checkbox label.
- |.e-checkbox-wrapper:hover .e-label|To customize the checkbox label on hover.
- |.e-checkbox-wrapper .e-frame.e-check|To customize the checked checkbox.
- |.e-checkbox-wrapper:hover .e-frame.e-check|To customize the checked checkbox when hover

Accessibility in React CheckBox component

The CheckBox component followed the accessibility guidelines and standards, including [ADA](#), [Section 508](#), [WCAG 2.2](#) standards, and [WCAG roles](#) that are commonly used to evaluate accessibility.

The accessibility compliance for the CheckBox component is outlined below.

| Accessibility Criteria | Compatibility |

| -- | -- |

| [WCAG 2.2](#) Support | |

| [Section 508](#) Support | |

| Screen Reader Support | |

| Right-To-Left Support | |

| Color Contrast | |

```

| Mobile Device Support |  |

| Keyboard Navigation Support |  |

| Accessibility Checker Validation |  |

| Axe-core Accessibility Validation |  |

<style>

.post .post-content img {
display: inline-block;
margin: 0.5em 0;
}

</style>

<div> - All
features of the component meet the requirement.</div>

<div> - Some features of the component do not meet the requirement.</div>

<div> - The component does not meet the requirement.</div>

```

WAI-ARIA attributes

The CheckBox component followed the [WAI-ARIA](#) patterns to meet the accessibility. The following ARIA attributes are used in the CheckBox component:

| Attributes | Purpose |

| --- | --- |

| `aria-disabled` | Indicates that the element is perceivable but disabled, so it is not editable or otherwise operable. |

Keyboard interaction

The CheckBox component followed the [keyboard interaction](#) guideline, making it easy for people who use assistive technologies (AT) and those who completely rely on keyboard navigation. The following keyboard shortcuts are supported by the CheckBox component.

| Press | To do this |

| --- | --- |

| Space | When the CheckBox has focus, pressing the Space key changes the state of the CheckBox. |

Ensuring accessibility

The CheckBox component's accessibility levels are ensured through an [accessibility-checker](#) and [axe-core](#) software tools during automated testing.

The accessibility compliance of the CheckBox component is shown in the following sample. Open the [sample](#) in a new window to evaluate the accessibility of the CheckBox component with accessibility tools.

See also

- [Accessibility in Syncfusion React components](#)

How To

Customized checkbox in React Check box component

Customize CheckBox Appearance

You can customize the appearance of the CheckBox component using the CSS rules. Define own CSS rules according to your requirement and assign the class name to the [cssClass](#) property.

The background and border color of the CheckBox is customized through custom classes to create primary, success, warning, danger, and info type of checkbox.

APP.JSX

```
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// To customize CheckBox appearance.
function App() {
    return (
        <ul>
            {/* Refer the 'e-primary' class details in 'style.css'.*/}
            <li><CheckBoxComponent label="Primary" cssClass="e-primary"
checked={true}/></li>
            {/* Refer the 'e-success' class details in 'style.css'.*/}
            <li><CheckBoxComponent label="Success" cssClass="e-success"
checked={true}/></li>
            {/* Refer the 'e-info' class details in 'style.css'.*/}
            <li><CheckBoxComponent label="Info" cssClass="e-info"
checked={true}/></li>
            {/* Refer the 'e-warning' class details in 'style.css'.*/}
            <li><CheckBoxComponent label="Warning" cssClass="e-warning"
checked={true}/></li>
            {/* Refer the 'e-danger' class details in 'style.css'.*/}
            <li><CheckBoxComponent label="Danger" cssClass="e-danger"
checked={true}/></li>
        </ul>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('check-box'));
```

APP.TSX

```
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// To customize CheckBox appearance.
function App() {
    return (
        <ul>
```

```

        { /* Refer the 'e-primary' class details in 'style.css'. */ }
        <li><CheckBoxComponent label="Primary" cssClass="e-primary"
checked={true} /></li>
        { /* Refer the 'e-success' class details in 'style.css'. */ }
        <li><CheckBoxComponent label="Success" cssClass="e-success"
checked={true} /></li>
        { /* Refer the 'e-info' class details in 'style.css'. */ }
        <li><CheckBoxComponent label="Info" cssClass="e-info"
checked={true} /></li>
        { /* Refer the 'e-warning' class details in 'style.css'. */ }
        <li><CheckBoxComponent label="Warning" cssClass="e-warning"
checked={true} /></li>
        { /* Refer the 'e-danger' class details in 'style.css'. */ }
        <li><CheckBoxComponent label="Danger" cssClass="e-danger"
checked={true} /></li>
    </ul>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('check-box'));

```

Custom Frame

CheckBox frame can be customized as per the requirement by adding CSS rules.

In the following example, to-do list is displayed with round checkbox by changing **border-radius** as **100%** by adding **e-custom** class.

APP.JSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
// To customize CheckBox appearance.
function App() {
    return (
        <ul>
            <li><CheckBoxComponent label="Buy Groceries" cssClass="e-custom"
checked={true}/></li>
            <li><CheckBoxComponent label="Pay Rent" cssClass="e-custom"/></li>
            <li><CheckBoxComponent label="Make Dinner" cssClass="e-
custom"/></li>
            <li><CheckBoxComponent label="Finish To-do List Article"
cssClass="e-custom"/></li>
        </ul>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('check-box'));

```

APP.TSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);

```

```
// To customize CheckBox appearance.
function App() {
  return (
    <ul>
      <li><CheckBoxComponent label="Buy Groceries" cssClass="e-custom"
checked={true} /></li>
      <li><CheckBoxComponent label="Pay Rent" cssClass="e-custom"
/></li>
      <li><CheckBoxComponent label="Make Dinner" cssClass="e-custom"
/></li>
      <li><CheckBoxComponent label="Finish To-do List Article"
cssClass="e-custom" /></li>
    </ul>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('check-box'));
```

Custom Check Icon

CheckBox check icon can be customized as per the requirement by adding CSS rules.

In the following example, the check icon can be customized by changing check icon content, background and border color in focus and hovered states by adding `e-checkicon` class.

APP.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To customize CheckBox appearance.
function App() {
  return (
    <ul>
      <li><CheckBoxComponent label="Buy Groceries" cssClass="e-
checkicon" checked={true} /></li>
      <li><CheckBoxComponent label="Pay Rent" cssClass="e-
checkicon" /></li>
      <li><CheckBoxComponent label="Make Dinner" cssClass="e-
checkicon" /></li>
      <li><CheckBoxComponent label="Finish To-do List Article"
cssClass="e-checkicon" /></li>
    </ul>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('check-box'));
```

APP.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To customize CheckBox appearance.
function App() {
```

```

    return (
      <ul>
        <li><CheckBoxComponent label="Buy Groceries" cssClass="e-checkicon" checked={true} /></li>
        <li><CheckBoxComponent label="Pay Rent" cssClass="e-checkicon" /></li>
        <li><CheckBoxComponent label="Make Dinner" cssClass="e-checkicon" /></li>
        <li><CheckBoxComponent label="Finish To-do List Article" cssClass="e-checkicon" /></li>
      </ul>
    );
  }
  export default App;
  ReactDOM.render(<App />, document.getElementById('check-box'));

```

Name and value in form submit in React Check box component

The [name](#) attribute of the CheckBox is used to group Checkboxes. When the Checkboxes are grouped in form, the checked items [value](#) attribute will post to the server on form submit which can be retrieved through the name. The disabled and unchecked CheckBox value will not be sent to the server on form submit.

In the following code snippet, Cricket and Hockey are in the checked state, Tennis is in [disabled](#) state and Basketball is in unchecked state. Now, the value that is in checked state only be sent on form submit.

APP.JSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ButtonComponent, CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
// Name and Value attribute in form submit.
function App() {
  return (<form>
    <ul>
      <li><CheckBoxComponent name="Sport" value="Cricket" label="Cricket" checked={true} /></li>
      <li><CheckBoxComponent name="Sport" value="Hockey" label="Hockey" checked={true} /></li>
      <li><CheckBoxComponent name="Sport" value="Tennis" label="Tennis" disabled={true} /></li>
      <li><CheckBoxComponent name="Sport" value="Basketball" label="Basketball" /></li>
      <li><ButtonComponent isPrimary={true}>Submit</ButtonComponent></li>
    </ul>
  </form>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('check-box'));

```

APP.TSX

```
import { enableRipple } from '@syncfusion/ej2-base';
import { ButtonComponent, CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
// Name and Value attribute in form submit.
function App() {
    return (
        <form>
            <ul>
                <li><CheckBoxComponent name="Sport" value="Cricket"
label="Cricket" checked={true} /></li>
                <li><CheckBoxComponent name="Sport" value="Hockey" label="Hockey"
checked={true} /></li>
                <li><CheckBoxComponent name="Sport" value="Tennis" label="Tennis"
disabled={true} /></li>
                <li><CheckBoxComponent name="Sport" value="Basketball"
label="Basketball" /></li>
                <li><ButtonComponent
isPrimary={true}>Submit</ButtonComponent></li>
            </ul>
        </form>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('check-box'));
```

Right to left in React Check box component

CheckBox component has RTL support. This can be achieved by setting [enableRtl](#) as `true`.

The following example illustrates how to enable right-to-left support in CheckBox component.

APP.JSX

```
import { enableRipple } from '@syncfusion/ej2-base';
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
// To customize CheckBox appearance.
function App() {
    return (
        <ul>
            <li><CheckBoxComponent label="Default" enableRtl={true}/></li>
        </ul>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('check-box'));
```

APP.TSX

```
import { enableRipple } from '@syncfusion/ej2-base';
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
```



```
// To customize CheckBox appearance.
function App() {
  return (
    <ul>
      <li><CheckBoxComponent label="Default" enableRtl={true} /></li>
    </ul>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('check-box'));
```

Ej1 api migration in React Check box component

This article describes the API migration process of Checkbox component from Essential JS 1 to Essential JS 2.

Properties

{% raw %}

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| Checkbox label | **Property:** *text*

 <EJ.CheckBox id="checkbox" text="Checkbox"></EJ.CheckBox> | **Property:** *label*

 <CheckBoxComponent id="checkbox" label="Checkbox"></CheckBoxComponent> |

| Checked state | **Property:** *enableTriState and checkState*

 <EJ.CheckBox id="checkbox" enableTriState={true} text="Checked state" checkState="check"></EJ.CheckBox> | **Property:** *checked*

 <CheckBoxComponent id="checkbox" checked={true} label="Checked state"></CheckBoxComponent> |

| Indeterminate state | **Property:** *enableTriState and checkState*

 <EJ.CheckBox id="checkbox" text="Indeterminate state" enableTriState={true} checkState="indeterminate"></EJ.CheckBox> | **Property:** *indeterminate*

 <CheckBoxComponent id="checkbox" indeterminate={true} label="Intermediate state"></CheckBoxComponent> |

| Adding custom css class | **Property:** *cssClass*

 <EJ.CheckBox id="checkbox" text="Checkbox" cssClass="custom-class"></EJ.CheckBox> | **Property:** *cssClass*

 <CheckBoxComponent id="checkbox" cssClass="custom-class" label="Checkbox"></CheckBoxComponent> |

| Label position | Not applicable | **Property:** *labelPosition*

 <CheckBoxComponent id="checkbox" label="Checkbox" labelPosition="Before"></CheckBoxComponent> |

| Disabled state | **Property:** *enabled*

 <EJ.CheckBox id="checkbox" text="Checkbox" enabled={false}></EJ.CheckBox> | **Property:** *disabled*

 <CheckBoxComponent id="checkbox" label="Checkbox" disabled={true}></CheckBoxComponent> |

| State persistence | **Property:** *enablePersistence*

 <EJ.CheckBox id="checkbox" text="Checkbox" enablePersistence={true}></EJ.CheckBox> | **Property:** *enablePersistence*

 <CheckBoxComponent id="checkbox" label="Checkbox" enablePersistence={true}></CheckBoxComponent> |

| RTL | **Property:** *enableRTL*

 <EJ.CheckBox id="checkbox" text="Checkbox" enableRTL={true}></EJ.CheckBox> | **Property:** *enableRtl*

 <CheckBoxComponent id="checkbox" label="Checkbox" enableRtl={true}></CheckBoxComponent> |

| HTML Attributes | **Property:** *htmlAttributes*

 var attributes = { required:"required" };
 <EJ.CheckBox id="checkbox" text="Checkbox" htmlAttributes={attributes}></EJ.CheckBox> | Not applicable |

| Id property | **Property:** *id*

 <EJ.CheckBox id="sync"></EJ.CheckBox> | Not applicable |

| Prefix value of Id | **Property:** *idPrefix*

 <EJ.CheckBox id="checkbox" text="Checkbox" id-prefix="react"></EJ.CheckBox> | Not applicable |

| Name attribute | **Property:** *name*

 <EJ.CheckBox id="checkbox" name="conformation"></EJ.CheckBox> | **Property:** *name*

 <CheckBoxComponent id="checkbox" name="conformation"></CheckBoxComponent> |

| Value attribute | **Property:** *value*

 <EJ.CheckBox id="checkbox" name="conformation" value="Received"></EJ.CheckBox> | **Property:** *value*

 <CheckBoxComponent id="checkbox" name="conformation" value="Received"></CheckBoxComponent> |

| Show rounded corner | **Property:** *showRoundedCorner*

 <EJ.CheckBox id="checkbox" text="Checkbox" showRoundedCorner={true}></EJ.CheckBox> | Not applicable |

| Size | **Property:** *size*

 <EJ.CheckBox id="checkbox" text="Small" size="small"></EJ.CheckBox> | **Property:** *cssClass*

 <CheckBoxComponent id="checkbox" cssClass="e-small" label="Checkbox"></CheckBoxComponent> |

| Validation rules | **Property:** *validationRules*

 var rules = { required: true };
 <EJ.CheckBox id="checkbox" validationRules={rules}></EJ.CheckBox> | Not applicable |

| Validation message | **Property:** *validationMessage*

 var rules = { required: true };
 var message = { required: "Required CheckBox value" };
 <EJ.CheckBox id="checkbox" validationRules={rules} validationMessage={message}></EJ.CheckBox> | Not applicable |

Methods

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| Destroy | **Method:** *destroy*

 <EJ.CheckBox id="checkbox" text="Checkbox"></EJ.CheckBox>
 var checkbox = \$('#checkbox').ejCheckBox('instance');
 checkbox.destroy(); | **Method:** *destroy*

 <CheckBoxComponent id="checkbox" label="Checkbox" ref={{(scope) => {this.checkbox = scope}}}></CheckBoxComponent>
 constructor(props: {}) {
 this.checkbox.destroy();
 } |

| Disable the Checkbox | **Method:** *disable*

 <EJ.CheckBox id="checkbox" text="Checkbox"></EJ.CheckBox>
 var checkbox = \$('#checkbox').ejCheckBox('instance');
 checkbox.disable(); | Not applicable |

| Enable the Checkbox | **Method:** *enable*

 <EJ.CheckBox id="checkbox" text="Checkbox"></EJ.CheckBox>
 var checkbox = \$('#checkbox').ejCheckBox('instance');
checkbox.enable(); | Not applicable |

| Check state of the Checkbox | **Method:** *isChecked*

 <EJ.CheckBox id="checkbox" text="Checkbox"></EJ.CheckBox>
 var checkbox = \$('#checkbox').ejCheckBox('instance');
checkbox.isChecked(); | Not applicable |

Events

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| BeforeChange Event | **Events:** *beforeChange*

 <EJ.CheckBox id="checkbox" text="Checkbox" beforeChange={beforeChange}></EJ.CheckBox>
function beforeChange(args) {
 / code block */
} | Not applicable |

| Change Event | **Events:** *change*

 <EJ.CheckBox id="checkbox" text="Checkbox" change={change}></EJ.CheckBox>
function change(args) {
 / code block /
} | **Events:** *change*

 <CheckBoxComponent id="checkbox" label="Checkbox" change={this.change.bind(this)}></CheckBoxComponent>
change(args) {
 / code block /
} |

| created Event | **Events:** *create*

 <EJ.CheckBox id="checkbox" text="Checkbox" create={create}></EJ.CheckBox>
function create(args) {
 / code block /
} | **Events:** *created*

 <CheckBoxComponent id="checkbox" label="Checkbox" created={this.created.bind(this)}></CheckBoxComponent>
created() {
 / code block /
} |

| Destroy Event | **Events:** *destroy*

 <EJ.CheckBox id="checkbox" text="Checkbox" destroy={destroy}></EJ.CheckBox>
function destroy(args) {
 / code block */
} | Not applicable |

{% endraw %}

Chips

Getting Started

This section explains how to create a simple Chip and to configure the Chip component.

Dependencies

The list of dependencies required to use the Chip component in your application is given below:

`javascript

|-- @syncfusion/ej2-react-buttons

|-- @syncfusion/ej2-base

|-- @syncfusion/ej2-buttons

|-- @syncfusion/ej2-react-base

,

Installation and Configuration

You can use [Create-react-app](#) to setup

the applications. To install `create-react-app` run the following command.

```
`bash
```

```
npm install -g create-react-app
```

```
`
```

To set-up a React application in TypeScript environment, run the following command.

```
`
```

```
npx create-react-app my-app --template typescript
```

```
cd my-app
```

```
npm start
```

```
`
```

To set-up a React application in JavaScript environment, run the following command.

```
`
```

```
npx create-react-app my-app
```

```
cd my-app
```

```
npm start
```

```
`
```

Adding Syncfusion packages

All the available Essential JS 2 packages are published in [npmjs.com](https://www.npmjs.com) public registry.

To install Chip component, use the following command

```
`bash
```

```
npm install @syncfusion/ej2-react-buttons --save
```

```
`
```

Adding CSS Reference

Import the Chip component's required CSS references as follows in `src/App.css`.

```
`css
```

```
@import "../node_modules/@syncfusion/ej2-react-buttons/styles/material.css";
```

```
`
```

Adding Chip component to the Application

To include the Chip component in your application import the `ChipListComponent` from `ej2-react-buttons` package in `App.tsx`.

Add the Chip component in application as shown in below code example.

```
`ts
```

```
import * as React from 'react';
import './App.css';
// Import the Button.
import { ChipListComponent } from '@syncfusion/ej2-react-buttons';
// To render Button.
function App() {
  return (
    <ChipListComponent text="Janet Leverling"></ChipListComponent>
  );
}
export default App;
```

Running the application

Run the application in the browser using the following command:

```
npm start
```

The following example shows a basic Chip component.

APP.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent } from '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
  return (<ChipListComponent text="Janet Leverling"></ChipListComponent>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));
```

APP.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent } from '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
  return (
    <ChipListComponent text="Janet Leverling"></ChipListComponent>
  );
};
```

```
}  
export default App;  
ReactDOM.render(<App />, document.getElementById('chip'));
```

Types in React Chips component

The ChipList control has the following types.

- Input Chip
- Choice Chip
- Filter Chip
- Action Chip

Input Chip

Input Chip holds information in compact form. It converts user input into chips.

APP.JSX

```
import * as React from 'react';  
import * as ReactDOM from 'react-dom';  
import { ChipListComponent, ChipsDirective, ChipDirective } from  
'@syncfusion/ej2-react-buttons';  
import { enableRipple } from '@syncfusion/ej2-base';  
enableRipple(true);  
// To render Chip.  
function App() {  
    return (<ChipListComponent id="chip-avatar" enableDelete={true}  
selection="Single">  
        <ChipsDirective>  
            <ChipDirective text="Andrew"></ChipDirective>  
            <ChipDirective text="Janet"></ChipDirective>  
            <ChipDirective text="Laura"></ChipDirective>  
            <ChipDirective text="Margaret"></ChipDirective>  
        </ChipsDirective>  
    </ChipListComponent>);  
}  
export default App;  
ReactDOM.render(<App />, document.getElementById('chip'));
```

APP.TSX

```
import * as React from 'react';  
import * as ReactDOM from 'react-dom';  
import { ChipListComponent, ChipsDirective, ChipDirective } from  
'@syncfusion/ej2-react-buttons';  
import { enableRipple } from '@syncfusion/ej2-base';  
enableRipple(true);  
// To render Chip.  
function App() {  
    return (  
        <ChipListComponent id="chip-avatar" enableDelete={true}  
selection="Single">  
            <ChipsDirective>  
                <ChipDirective text="Andrew"></ChipDirective>  
                <ChipDirective text="Janet"></ChipDirective>
```

```

        <ChipDirective text="Laura"></ChipDirective>
        <ChipDirective text="Margaret"></ChipDirective>
    </ChipsDirective>
</ChipListComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));

```

Choice Chip

Choice Chip allows you to select a single chip from the set of ChipList/ChipCollection. It can be enabled by setting the `selection` property to `Single`.

APP.JSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
'@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
    return (<ChipListComponent id="chip-avatar" selection="Single">
        <ChipsDirective>
            <ChipDirective text="Small"></ChipDirective>
            <ChipDirective text="Medium"></ChipDirective>
            <ChipDirective text="Large"></ChipDirective>
            <ChipDirective text="Extra Large"></ChipDirective>
        </ChipsDirective>
    </ChipListComponent>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));

```

APP.TSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
'@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
    return (
        <ChipListComponent id="chip-avatar" selection="Single">
            <ChipsDirective>
                <ChipDirective text="Small"></ChipDirective>
                <ChipDirective text="Medium"></ChipDirective>
                <ChipDirective text="Large"></ChipDirective>
                <ChipDirective text="Extra Large"></ChipDirective>
            </ChipsDirective>
        </ChipListComponent>
    );
}

```

```
}  
export default App;  
ReactDOM.render(<App />, document.getElementById('chip'));
```

Filter Chip

Filter Chip allows you to select a multiple chip from the set of ChipList/ChipCollection. It can be enabled by setting the `selection` property to `Multiple`.

APP.JSX

```
import * as React from 'react';  
import * as ReactDOM from 'react-dom';  
import { ChipListComponent, ChipsDirective, ChipDirective } from  
'@syncfusion/ej2-react-buttons';  
import { enableRipple } from '@syncfusion/ej2-base';  
enableRipple(true);  
// To render Chip.  
function App() {  
    return (<ChipListComponent id="chip-avatar" selection="Multiple">  
        <ChipsDirective>  
            <ChipDirective text="Chai"></ChipDirective>  
            <ChipDirective text="Chung"></ChipDirective>  
            <ChipDirective text="Aniseed Syrup"></ChipDirective>  
            <ChipDirective text="Ikura"></ChipDirective>  
        </ChipsDirective>  
    </ChipListComponent>);  
}  
export default App;  
ReactDOM.render(<App />, document.getElementById('chip'));
```

APP.TSX

```
import * as React from 'react';  
import * as ReactDOM from 'react-dom';  
import { ChipListComponent, ChipsDirective, ChipDirective } from  
'@syncfusion/ej2-react-buttons';  
import { enableRipple } from '@syncfusion/ej2-base';  
enableRipple(true);  
// To render Chip.  
function App() {  
    return (  
        <ChipListComponent id="chip-avatar" selection="Multiple">  
            <ChipsDirective>  
                <ChipDirective text="Chai"></ChipDirective>  
                <ChipDirective text="Chung"></ChipDirective>  
                <ChipDirective text="Aniseed Syrup"></ChipDirective>  
                <ChipDirective text="Ikura"></ChipDirective>  
            </ChipsDirective>  
        </ChipListComponent>  
    );  
}  
export default App;  
ReactDOM.render(<App />, document.getElementById('chip'));
```


Action Chip

The Action Chip triggers the event like click or delete, which helps doing action based on the event.

APP.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
 '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
    function chipClick(e) {
        alert('you have clicked ' + e.target.textContent);
    }
    return (<ChipListComponent id="chip-avatar"
onClick={chipClick.bind(this)}>
        <ChipsDirective>
            <ChipDirective text="Send a text"/>
            <ChipDirective text="Set a remainder"/>
            <ChipDirective text="Read my emails"/>
            <ChipDirective text="Set alarm"/>
        </ChipsDirective>
    </ChipListComponent>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));
```

APP.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
 '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
    function chipClick(e: any) {
        alert('you have clicked ' + e.target.textContent);
    }
    return (
        <ChipListComponent id="chip-avatar" onClick={chipClick.bind(this)}>
            <ChipsDirective>
                <ChipDirective text="Send a text" />
                <ChipDirective text="Set a remainder" />
                <ChipDirective text="Read my emails" />
                <ChipDirective text="Set alarm" />
            </ChipsDirective>
        </ChipListComponent>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));
```

Deletable Chip

Deletable Chip allows you to delete a chip from ChipList/ChipCollection. It can be enabled by setting the `enableDelete` property to `true`.

APP.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
 '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
    function chipClick(e) {
        alert('you have clicked ' + e.target.textContent);
    }
    return (<ChipListComponent id="chip-avatar" enableDelete={true}>
        <ChipsDirective>
            <ChipDirective text="Send a text"></ChipDirective>
            <ChipDirective text="Set a remainder"></ChipDirective>
            <ChipDirective text="Read my emails"></ChipDirective>
            <ChipDirective text="Set alarm"></ChipDirective>
        </ChipsDirective>
    </ChipListComponent>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));
```

APP.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
 '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
    function chipClick(e: any) {
        alert('you have clicked ' + e.target.textContent);
    }
    return (
        <ChipListComponent id="chip-avatar" enableDelete={true}>
            <ChipsDirective>
                <ChipDirective text="Send a text"></ChipDirective>
                <ChipDirective text="Set a remainder"></ChipDirective>
                <ChipDirective text="Read my emails"></ChipDirective>
                <ChipDirective text="Set alarm"></ChipDirective>
            </ChipsDirective>
        </ChipListComponent>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));
```

Customization in React Chips component

This section explains the customization of styles, leading icons, avatar, and trailing icons in Chip control.

Styles

The Chip control has the following predefined styles that can be defined using the `cssClass` property.

Class	Description
-----	-----
e-primary	Represents a primary chip.
e-success	Represents a positive chip.
e-info	Represents an informative chip.
e-warning	Represents a chip with caution.
e-danger	Represents a negative chip.

APP.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
 '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
    return (<ChipListComponent id="chip-avatar">
        <ChipsDirective>
            <ChipDirective text="Primary" cssClass="e-
primary"></ChipDirective>
            <ChipDirective text="Success" cssClass="e-
success"></ChipDirective>
            <ChipDirective text="Info" cssClass="e-info"></ChipDirective>
            <ChipDirective text="Warning" cssClass="e-
warning"></ChipDirective>
            <ChipDirective text="Danger" cssClass="e-
danger"></ChipDirective>
        </ChipsDirective>
    </ChipListComponent>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));
```

APP.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
 '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
    return (
```

```

    <ChipListComponent id="chip-avatar">
      <ChipsDirective>
        <ChipDirective text="Primary" cssClass="e-
primary"></ChipDirective>
        <ChipDirective text="Success" cssClass="e-
success"></ChipDirective>
        <ChipDirective text="Info" cssClass="e-info"></ChipDirective>
        <ChipDirective text="Warning" cssClass="e-
warning"></ChipDirective>
        <ChipDirective text="Danger" cssClass="e-
danger"></ChipDirective>
      </ChipsDirective>
    </ChipListComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));

```

Leading Icon

You can add and customize the leading icon of chip using the `leadingIconCss` property.

APP.JSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
 '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
  return (<ChipListComponent id="chip-avatar">
    <ChipsDirective>
      <ChipDirective text="Andrew"
leadingIconCss='andrew'></ChipDirective>
      <ChipDirective text="Janet"
leadingIconCss='janet'></ChipDirective>
      <ChipDirective text="Laura"
leadingIconCss='laura'></ChipDirective>
      <ChipDirective text="Margaret"
leadingIconCss='margaret'></ChipDirective>
    </ChipsDirective>
  </ChipListComponent>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));

```

APP.TSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
 '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);

```

```
// To render Chip.
function App() {
  return (
    <ChipListComponent id="chip-avatar">
      <ChipsDirective>
        <ChipDirective text="Andrew"
leadingIconCss='andrew'></ChipDirective>
        <ChipDirective text="Janet"
leadingIconCss='janet'></ChipDirective>
        <ChipDirective text="Laura"
leadingIconCss='laura'></ChipDirective>
        <ChipDirective text="Margaret"
leadingIconCss='margaret'></ChipDirective>
      </ChipsDirective>
    </ChipListComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));
```

Avatar

You can add and customize the avatar of chip using the `avatarIconCss` property.

APP.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
 '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
  return (<ChipListComponent id="chip-avatar">
    <ChipsDirective>
      <ChipDirective text="Andrew"
avatarIconCss='andrew'></ChipDirective>
      <ChipDirective text="Janet"
avatarIconCss='janet'></ChipDirective>
      <ChipDirective text="Laura"
avatarIconCss='laura'></ChipDirective>
      <ChipDirective text="Margaret"
avatarIconCss='margaret'></ChipDirective>
    </ChipsDirective>
  </ChipListComponent>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));
```

APP.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
 '@syncfusion/ej2-react-buttons';
```

```
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
    return (
        <ChipListComponent id="chip-avatar">
            <ChipsDirective>
                <ChipDirective text="Andrew"
avatarIconCss='andrew'></ChipDirective>
                <ChipDirective text="Janet"
avatarIconCss='janet'></ChipDirective>
                <ChipDirective text="Laura"
avatarIconCss='laura'></ChipDirective>
                <ChipDirective text="Margaret"
avatarIconCss='margaret'></ChipDirective>
            </ChipsDirective>
        </ChipListComponent>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));
```

Avatar Content

You can add and customize the avatar content of chip using the `avatarText` property.

APP.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
 '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
    return (<ChipListComponent id="chip-avatar">
        <ChipsDirective>
            <ChipDirective text="Andrew" avatarText='A'></ChipDirective>
            <ChipDirective text="Janet" avatarText='J'></ChipDirective>
            <ChipDirective text="Laura" avatarText='L'></ChipDirective>
            <ChipDirective text="Margaret" avatarText='M'></ChipDirective>
        </ChipsDirective>
    </ChipListComponent>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));
```

APP.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
 '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
```

```
// To render Chip.
function App() {
  return (
    <ChipListComponent id="chip-avatar">
      <ChipsDirective>
        <ChipDirective text="Andrew" avatarText='A'></ChipDirective>
        <ChipDirective text="Janet" avatarText='J'></ChipDirective>
        <ChipDirective text="Laura" avatarText='L'></ChipDirective>
        <ChipDirective text="Margaret" avatarText='M'></ChipDirective>
      </ChipsDirective>
    </ChipListComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));
```

Trailing Icon

You can add and customize the trailing icon of chip using the `trailingIconCss` property.

APP.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
 '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
  return (<ChipListComponent id="chip-avatar">
    <ChipsDirective>
      <ChipDirective text="Andrew" trailingIconCss='e-dlt-
btn'></ChipDirective>
      <ChipDirective text="Janet" trailingIconCss='e-dlt-
btn'></ChipDirective>
      <ChipDirective text="Laura" trailingIconCss='e-dlt-
btn'></ChipDirective>
      <ChipDirective text="Margaret" trailingIconCss='e-dlt-
btn'></ChipDirective>
    </ChipsDirective>
  </ChipListComponent>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));
```

APP.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
 '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
```

```

return (
  <ChipListComponent id="chip-avatar">
    <ChipsDirective>
      <ChipDirective text="Andrew" trailingIconCss= 'e-dlt-
btn'></ChipDirective>
      <ChipDirective text="Janet" trailingIconCss= 'e-dlt-
btn'></ChipDirective>
      <ChipDirective text="Laura" trailingIconCss= 'e-dlt-
btn'></ChipDirective>
      <ChipDirective text="Margaret" trailingIconCss= 'e-dlt-
btn'></ChipDirective>
    </ChipsDirective>
  </ChipListComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));

```

Outline Chip

Outline chip has the border with the background transparent. It can be set using the `cssClass` property.

APP.JSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
 '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
  return (<div>
    <ChipListComponent id="chip-avatar" cssClass="e-outline">
      <ChipsDirective>
        <ChipDirective text="Chai"></ChipDirective>
        <ChipDirective text="Chang"></ChipDirective>
        <ChipDirective text="Aniseed Syrup"></ChipDirective>
        <ChipDirective text="Ikura"></ChipDirective>
      </ChipsDirective>
    </ChipListComponent>
    <ChipListComponent id="chip-avatar" cssClass="e-outline"
enableDelete={true}>
      <ChipsDirective>
        <ChipDirective text="Andrew"></ChipDirective>
        <ChipDirective text="Janet"></ChipDirective>
        <ChipDirective text="Laura"></ChipDirective>
        <ChipDirective text="Margaret"></ChipDirective>
      </ChipsDirective>
    </ChipListComponent>
  </div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));

```

APP.TSX


```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ChipListComponent, ChipsDirective, ChipDirective } from
 '@syncfusion/ej2-react-buttons';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
// To render Chip.
function App() {
  return (
    <div>
      <ChipListComponent id="chip-avatar" cssClass="e-outline">
        <ChipsDirective>
          <ChipDirective text="Chai"></ChipDirective>
          <ChipDirective text="Chang"></ChipDirective>
          <ChipDirective text="Aniseed Syrup"></ChipDirective>
          <ChipDirective text="Ikura"></ChipDirective>
        </ChipsDirective>
      </ChipListComponent>
      <ChipListComponent id="chip-avatar" cssClass="e-outline"
enableDelete={true}>
        <ChipsDirective>
          <ChipDirective text="Andrew"></ChipDirective>
          <ChipDirective text="Janet"></ChipDirective>
          <ChipDirective text="Laura"></ChipDirective>
          <ChipDirective text="Margaret"></ChipDirective>
        </ChipsDirective>
      </ChipListComponent>
    </div>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('chip'));

```

Style in React Chips component

The following content provides the exact CSS structure that can be used to modify the control's appearance based on the user preference.

Customizing the chip text

Use the following CSS to customize the chip text properties.

```

`css
.e-chip .e-chip-text {
font-size: 20px;
color: black;
font-weight: normal;
}
`

```

Customizing the chip icon

Use the following CSS to customize the chip icon properties.

```
`css
.e-chip .e-icon {
background-image: url('https://ej2.syncfusion.com/demos/src/chips/images/laura.png');
opacity: 0.8;
}
`
```

Customizing the chip delete button

Use the following CSS to customize the chip delete button.

```
`css
.e-chip-list .e-chip .e-chip-delete.e-dlt-btn {
color: #e3165b;
font-size: 12px;
}
`
```

Customizing the chip outline

Use the following CSS to customize the chip outline.

```
`css
.e-chip-list .e-chip.e-outline {
border-color: #e3165b;
border-width: 3px;
}
`
```

Customizing the chip on selection

Use the following CSS to customize the chip on selection.

```
`css
/ To customize single chip on selection /
.e-chip-list.e-selection .e-chip.e-active {
background-color: #ffca1c;
color: #e3165b;
}
/ To customize multiple chip on selection /
.e-chip-list .e-chip.e-active {
background-color: #e3165b;
color: white;
}
```

```
}  
`
```

Customizing the chip avatar text

Use the following CSS to customize the chip avatar text properties.

```
`css  
.e-chip-list .e-chip .e-chip-avatar {  
background-color: #d51a1a;  
color: #fafafa;  
}  
`
```

Accessibility in React Chips component

The Chips component followed the accessibility guidelines and standards, including [ADA, Section 508](#), [WCAG 2.2](#) standards, and [WCAG roles](#) that are commonly used to evaluate accessibility.

The accessibility compliance for the Chips component is outlined below.

| Accessibility Criteria | Compatibility |

| -- | -- |

| [WCAG 2.2](#) Support | |

| [Section 508](#) Support | |

| Screen Reader Support | |

| Right-To-Left Support | |

| Color Contrast | |

| Mobile Device Support | |

| Keyboard Navigation Support | |

| [Accessibility Checker](#) Validation | |

| [Axe-core](#) Accessibility Validation | |

<style>

```
.post .post-content img {  
display: inline-block;
```

```
margin: 0.5em 0;
```

```
}
```

```
</style>
```

```
<div> - All features of the component meet the requirement.</div>
```

```
<div> - Some features of the component do not meet the requirement.</div>
```

```
<div> - The component does not meet the requirement.</div>
```

WAI-ARIA attributes

The Chips component followed the [WAI-ARIA](#) patterns to meet the accessibility. The following ARIA attributes are used in the Chips component:

Attributes	Purpose
------------	---------

---	---
-----	-----

<code>role=checkbox</code>	Indicates the ChipList component wrapper element as <code>checkbox</code> .
----------------------------	---

<code>role=option</code>	Used to convey a significant and contextual message to the user(ChipList).
--------------------------	--

<code>role=button</code>	Used to convey a significant and contextual message to the user(Single Chip).
--------------------------	---

<code>aria-label</code>	Provides an accessible name for the Chip.
-------------------------	---

<code>aria-selected</code>	Indicates the element is selected.
----------------------------	------------------------------------

<code>aria-disabled</code>	Indicates element is perceivable but disabled.
----------------------------	--

<code>aria-multiselectable</code>	Indicates multiple items to be selected.
-----------------------------------	--

Keyboard interaction

The Chips component followed the [keyboard interaction](#) guideline, making it easy for people who use assistive technologies (AT) and those who completely rely on keyboard navigation. The following keyboard shortcuts are supported by the Chips component.

Keyboard shortcuts	Actions
--------------------	---------

-----	-----
-------	-------

Enter / Space	Selects the targeted chip from the ChipList/ChipCollection.
----------------------	--

Delete / Backspace	Deletes the targeted chip from the ChipList/ChipCollection.
---------------------------	--

Ensuring accessibility

The Chips component's accessibility levels are ensured through an [accessibility-checker](#) and [axe-core](#) software tools during automated testing.

The accessibility compliance of the Chips component is shown in the following sample. Open the [sample](#) in a new window to evaluate the accessibility of the Chips component with accessibility tools.

See also

- [Accessibility in Syncfusion React components](#)

3D Circular Chart

<!-- markdownlint-disable MD036 -->

Getting started

This section explains you the steps required to create a simple 3D Circular Chart and demonstrate the basic usage of the 3D Circular Chart control.

Dependencies

Below is the list of minimum dependencies required to use the 3D Circular Chart component.

```
`javascript
|-- @syncfusion/ej2-react-charts
|-- @syncfusion/ej2-base
|-- @syncfusion/ej2-data
|-- @syncfusion/ej2-charts
|-- @syncfusion/ej2-react-base
|-- @syncfusion/ej2-pdf-export
|-- @syncfusion/ej2-file-utils
|-- @syncfusion/ej2-compression
|-- @syncfusion/ej2-svg-base
`,`
```

Installation and configuration

You can use [create-react-app](#) to setup the applications.

To install `create-react-app` run the following command.

```
`npm install -g create-react-app
`,`
```

- To set-up a React application in TypeScript environment, run the following command.

```
`create-react-app quickstart --template typescript
cd quickstart
npm start
`,`
```

- To set-up a React application in JavaScript environment, run the following command.

,

```
create-react-app quickstart
```

```
cd quickstart
```

```
npm start
```

,

- Install Syncfusion packages using below command.

,

```
npm install @syncfusion/ej2-react-charts --save
```

,

Add 3D Circular Chart to the project

Now, you can start adding 3D Circular Chart component in the application.

For getting started, add the 3D Circular Chart component in `src/App.tsx` file using following code.

INDEX.JSX

```
import { CircularChart3DComponent } from '@syncfusion/ej2-react-charts';
import * as React from 'react';
function App() {
  return (<CircularChart3DComponent />);
}
export default App;
```

INDEX.TSX

```
import { CircularChart3DComponent } from '@syncfusion/ej2-react-charts';
import * as React from 'react';
function App() {
  return (<CircularChart3DComponent />);
}
export default App;
```

Pie Series

By default, the pie series will be rendered when assigning the JSON data to the series using the `dataSource` property. Map the field names in the JSON data to the `xName` and `yName` properties of the series.

INDEX.JSX

```
{% raw %}
import { CircularChart3DComponent, CircularChart3DSeriesCollectionDirective,
CircularChart3DSeriesDirective, PieSeries3D, CircularChartDataLabel3D,
CircularChartLegend3D, Inject } from '@syncfusion/ej2-react-charts';
import * as React from "react";
import * as ReactDOM from "react-dom";
```

```
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 }
  ];

  return <CircularChart3DComponent id='charts' title='Browser Market
Shares in November 2023' tilt={-45} legendSettings={{ visible: true,
position: 'Right' }}>
    <Inject services={[PieSeries3D, CircularChartDataLabel3D,
CircularChartLegend3D]}/>
    <CircularChart3DSeriesCollectionDirective>
      <CircularChart3DSeriesDirective dataSource={circularData}
xName='x' yName='y' dataLabel={{ visible: true, position: 'Outside', name:
'x', font: { fontWeight: '600' }, connectorStyle: { length: '40px' } }}>
        </CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>;
}
;
export default App;
ReactDOM.render(<App />, document.getElementById("charts"));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import { CircularChart3DComponent, CircularChart3DSeriesCollectionDirective,
CircularChart3DSeriesDirective, PieSeries3D, CircularChartDataLabel3D,
CircularChartLegend3D, Inject } from '@syncfusion/ej2-react-charts';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 }
  ];
  return <CircularChart3DComponent id='charts' title='Browser Market Shares
in November 2023' tilt={-45} legendSettings={{ visible: true, position:
'Right' }}>
    <Inject services={[PieSeries3D, CircularChartDataLabel3D,
CircularChartLegend3D]}/>
    <CircularChart3DSeriesCollectionDirective>
      <CircularChart3DSeriesDirective dataSource={circularData}
xName='x' yName='y' dataLabel={{ visible: true, position: 'Outside', name:
'x', font: { fontWeight: '600' }, connectorStyle: { length: '40px' } }}>
        </CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  
```

```
};  
export default App;  
ReactDOM.render(<App />, document.getElementById("charts"));  
{% endraw %}
```

Now run the `npm start` command in the console, it will run your application and open the browser window.

,

`npm start`

,

Pie and Donut in React 3D Circular Chart component

Pie chart

To render a pie series, inject the `PieSeries3D` into the service.

INDEX.JSX

```
{% raw %}  
import {  
  CircularChart3DComponent,  
  CircularChart3DSeriesCollectionDirective,  
  CircularChart3DSeriesDirective,  
  PieSeries3D,  
  CircularChartDataLabel3D,  
  Inject,  
} from '@syncfusion/ej2-react-charts';  
import * as React from 'react';  
import * as ReactDOM from 'react-dom';  
function App() {  
  const circularData = [  
    { x: 'Chrome', y: 62.92 },  
    { x: 'Internet Explorer', y: 6.12 },  
    { x: 'Opera', y: 3.15 },  
    { x: 'Edge', y: 5.5 },  
    { x: 'Safari', y: 19.97 },  
    { x: 'Others', y: 2.34 },  
  ];  
  return (  
    <CircularChart3DComponent  
      id="charts"  
      title="Browser Market Shares in November 2023"  
      tilt={-45}  
    >  
      <Inject  
        services={[  
          PieSeries3D,  
          CircularChartDataLabel3D,  
        ]}  
      />  
      <CircularChart3DSeriesCollectionDirective>  
        <CircularChart3DSeriesDirective  
          dataSource={circularData}  
          xName="x"  

```



```

        yName="y"
      ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
  </CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));

```

```
{% endraw %}
```

Radius customization

By default, the radius of the pie series will be 80% of the size, which is the minimum of the 3D Circular Chart's width and height. You can customize this by using the `radius` property of the series.

INDEX.JSX

```
{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          radius='80%'
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          radius='80%'
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}
```

Various radius pie chart

You can assign different radii to each slice of the pie by fetching the radius from the data source and using it with the `radius` property in the `series`.

INDEX.JSX

```
{% raw %}
```

```

import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Belgium', y: 551500, r: '110.7', text: 'Belgium' },
    { x: 'Dominican Republic', y: 312685, r: '137.5', text: 'Dominican Republic' },
    { x: 'Cuba', y: 350000, r: '124.6', text: 'Cuba' },
    { x: 'Egypt', y: 301000, r: '150.8', text: 'Egypt' },
    { x: 'Kazakhstan', y: 300000, r: '155.5', text: 'Kazakhstan' },
    { x: 'Somalia', y: 357022, r: '160.6', text: 'Somalia' },
    { x: 'Argentina', y: 505370, r: '100', text: 'Argentina' },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          radius='r'
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,

```

```

CircularChartDataLabel3D,
Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Belgium', y: 551500, r: '110.7', text: 'Belgium' },
    { x: 'Dominican Republic', y: 312685, r: '137.5', text: 'Dominican Republic' },
    { x: 'Cuba', y: 350000, r: '124.6', text: 'Cuba' },
    { x: 'Egypt', y: 301000, r: '150.8', text: 'Egypt' },
    { x: 'Kazakhstan', y: 300000, r: '155.5', text: 'Kazakhstan' },
    { x: 'Somalia', y: 357022, r: '160.6', text: 'Somalia' },
    { x: 'Argentina', y: 505370, r: '100', text: 'Argentina' },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          radius='r'
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Donut chart

To achieve a donut in the pie series, customize the `innerRadius` property of the series. By setting a value greater than 0%, a donut will appear. The `innerRadius` property takes value from 0% to 100% of the pie radius.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,

```

```

CircularChartDataLabel3D,
Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          innerRadius='40%'
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [

```

```

    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          innerRadius='40%'
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Text and fill color mapping

The text and the fill color from the data source can be mapped to the 3D Circular Chart using `pointColorMapping` in the series and `name` in the data label, respectively.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92, text: 'Chrome: 35%', color: 'yellow' },
    {
      x: 'Internet Explorer',

```

```

        y: 6.12,
        text: 'Internet Explorer: 15%',
        color: 'orange',
      },
      { x: 'Opera', y: 3.15, text: 'Opera: 10%', color: 'red' },
      { x: 'Edge', y: 5.5, text: 'Edge: 15%', color: 'green' },
      { x: 'Safari', y: 19.97, text: 'Safari: 20%', color: 'blue' },
      { x: 'Others', y: 2.34, text: 'Others: 5%', color: 'black' },
    ];
    return (
      <CircularChart3DComponent
        id="charts"
        title="Browser Market Shares in November 2023"
        tilt={-45}
      >
        <Inject services={[PieSeries3D, CircularChartDataLabel3D]} />
        <CircularChart3DSeriesCollectionDirective>
          <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            radius="80%"
            innerRadius="40%"
            pointColorMapping='color'
            dataLabel={{ visible: true, position: 'Outside', name: 'text' }}
          ></CircularChart3DSeriesDirective>
        </CircularChart3DSeriesCollectionDirective>
      </CircularChart3DComponent>
    );
  }
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92, text: 'Chrome: 35%', color: 'yellow' },
    {
      x: 'Internet Explorer',
      y: 6.12,
      text: 'Internet Explorer: 15%',
      color: 'orange',
    },
  ],

```



```

    { x: 'Opera', y: 3.15, text: 'Opera: 10%', color: 'red' },
    { x: 'Edge', y: 5.5, text: 'Edge: 15%', color: 'green' },
    { x: 'Safari', y: 19.97, text: 'Safari: 20%', color: 'blue' },
    { x: 'Others', y: 2.34, text: 'Others: 5%', color: 'black' },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
    >
      <Inject services={[PieSeries3D, CircularChartDataLabel3D]} />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          radius="80%"
          innerRadius="40%"
          pointColorMapping='color'
          dataLabel={{ visible: true, position: 'Outside', name: 'text' }}
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Customization

Individual points in pie chart can be customized using the `pointRender` event.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92, text: 'Chrome: 40%' },
    { x: 'Internet Explorer', y: 6.12, text: 'Chrome: 15%' },
    { x: 'Opera', y: 3.15, text: 'Chrome: 8%' },
    { x: 'Edge', y: 5.5, text: 'Chrome: 15%' },
    { x: 'Safari', y: 19.97, text: 'Chrome: 20%' },
    { x: 'Others', y: 2.34, text: 'Chrome: 2%' },
  ];
  const textRender = function (args) {

```

```

    if (args.point.index === 0) {
      args.fill = 'red';
    }
  };
  return (
    <CircularChart3DComponent
      id="charts"
      pointRender={textRender}
      title="Browser Market Shares in November 2023"
      tilt={-45}
    >
      <Inject services={[PieSeries3D, CircularChartDataLabel3D]} />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          dataLabel={{
            visible: true,
            enableRotation: true,
          }}
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92, text: 'Chrome: 40%' },
    { x: 'Internet Explorer', y: 6.12, text: 'Chrome: 15%' },
    { x: 'Opera', y: 3.15, text: 'Chrome: 8%' },
    { x: 'Edge', y: 5.5, text: 'Chrome: 15%' },
    { x: 'Safari', y: 19.97, text: 'Chrome: 20%' },
    { x: 'Others', y: 2.34, text: 'Chrome: 2%' },
  ];
  const textRender = function (args) {
    if (args.point.index === 0) {
      args.fill = 'red';
    }
  }
}

```

```

};
return (
  <CircularChart3DComponent
    id="charts"
    pointRender={textRender}
    title="Browser Market Shares in November 2023"
    tilt={-45}
  >
    <Inject services={[PieSeries3D, CircularChartDataLabel3D]} />
    <CircularChart3DSeriesCollectionDirective>
      <CircularChart3DSeriesDirective
        dataSource={circularData}
        xName="x"
        yName="y"
        dataLabel={{
          visible: true,
          enableRotation: true,
        }}
      ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
  </CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% enddraw %}

```

Data Label in React 3D Circular Chart component

A data label refers to a label associated with specific data points. It can be added to a 3D Circular Chart series by enabling the `visible` option in the `dataLabel` property. By default, the labels will arrange themselves smartly to avoid overlapping.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent

```

```

        id="charts"
        title="Browser Market Shares in November 2023"
        tilt={-45}
        legendSettings={{ visible: true, position: 'Right' }}
    >
    <Inject
        services={[
            PieSeries3D,
            CircularChartDataLabel3D,
        ]}
    />
    <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            dataLabel={{
                visible: true
            }}
        ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartDataLabel3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData: any[] = [
        { x: 'Chrome', y: 62.92 },
        { x: 'Internet Explorer', y: 6.12 },
        { x: 'Opera', y: 3.15 },
        { x: 'Edge', y: 5.5 },
        { x: 'Safari', y: 19.97 },
        { x: 'Others', y: 2.34 },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Browser Market Shares in November 2023"
            tilt={-45}
            legendSettings={{ visible: true, position: 'Right' }}

```

```

    >
    <Inject
      services={[
        PieSeries3D,
        CircularChartDataLabel3D,
      ]}
    />
    <CircularChart3DSeriesCollectionDirective>
      <CircularChart3DSeriesDirective
        dataSource={circularData}
        xName="x"
        yName="y"
        dataLabel={{
          visible: true
        }}
      ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
  </CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% enddraw %}

```

To use the data label feature, inject the `CircularChartDataLabel3D` module into the services.

Positioning

Using the `position` property, we can place the data label either `inside` or `outside` the 3D Circular Chart.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
    >

```

```

        legendSettings={{ visible: true, position: 'Right' }}
    >
    <Inject
        services={[
            PieSeries3D,
            CircularChartDataLabel3D,
        ]}
    />
    <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            dataLabel={{
                visible: true,
                position: 'Outside'
            }}
        ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartDataLabel3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData: any[] = [
        { x: 'Chrome', y: 62.92 },
        { x: 'Internet Explorer', y: 6.12 },
        { x: 'Opera', y: 3.15 },
        { x: 'Edge', y: 5.5 },
        { x: 'Safari', y: 19.97 },
        { x: 'Others', y: 2.34 },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Browser Market Shares in November 2023"
            tilt={-45}
            legendSettings={{ visible: true, position: 'Right' }}
        >
            <Inject

```

```

        services={ [
            PieSeries3D,
            CircularChartDataLabel3D,
        ] }
    />
    <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            dataLabel={{
                visible: true,
                position: 'Outside'
            }}
        ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% enddraw %}

```

Data label template

The label content can be formatted using the template option. Inside the template, placeholder text `${point.x}` and `${point.y}` can be added to display the corresponding data point's x & y value. The data label template can be set using the `template` property.

INDEX.JSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartDataLabel3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData = [
        { x: 'Chrome', y: 62.92, text: 'Chrome: 40%' },
        { x: 'Internet Explorer', y: 6.12, text: 'Chrome: 15%' },
        { x: 'Opera', y: 3.15, text: 'Chrome: 8%' },
        { x: 'Edge', y: 5.5, text: 'Chrome: 15%' },
        { x: 'Safari', y: 19.97, text: 'Chrome: 20%' },
        { x: 'Others', y: 2.34, text: 'Chrome: 2%' },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Browser Market Shares in November 2023"
            tilt={-45}
            legendSettings={{ visible: true, position: 'Right' }}

```

```

    >
    <Inject
      services={[
        PieSeries3D,
        CircularChartDataLabel3D,
      ]}
    />
    <CircularChart3DSeriesCollectionDirective>
      <CircularChart3DSeriesDirective
        dataSource={circularData}
        xName="x"
        yName="y"
        dataLabel={{
          visible: true,
          position: 'Outside',
          template: '${point.x}: ${point.y}'
        }}
      ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
  </CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92, text: 'Chrome: 40%' },
    { x: 'Internet Explorer', y: 6.12, text: 'Chrome: 15%' },
    { x: 'Opera', y: 3.15, text: 'Chrome: 8%' },
    { x: 'Edge', y: 5.5, text: 'Chrome: 15%' },
    { x: 'Safari', y: 19.97, text: 'Chrome: 20%' },
    { x: 'Others', y: 2.34, text: 'Chrome: 2%' },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      legendSettings={{ visible: true, position: 'Right' }}
    >
      <Inject

```



```

        services={[
            PieSeries3D,
            CircularChartDataLabel3D,
        ]}
    />
    <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            dataLabel={{
                visible: true,
                position: 'Outside',
                template: '${point.x}: ${point.y}'
            }}
        ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% enddraw %}

```

Connector line

The connector line will be visible when the data label is placed outside the chart. It can be customized using properties such as **color**, **width**, **length**, and **dashArray** within the **connectorStyle** property.

INDEX.JSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartDataLabel3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData = [
        { x: 'Chrome', y: 62.92, text: 'Chrome: 40%' },
        { x: 'Internet Explorer', y: 6.12, text: 'Chrome: 15%' },
        { x: 'Opera', y: 3.15, text: 'Chrome: 8%' },
        { x: 'Edge', y: 5.5, text: 'Chrome: 15%' },
        { x: 'Safari', y: 19.97, text: 'Chrome: 20%' },
        { x: 'Others', y: 2.34, text: 'Chrome: 2%' },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Browser Market Shares in November 2023"
            tilt={-45}
            legendSettings={{ visible: true, position: 'Right' }}

```

```

>
<Inject services={[PieSeries3D, CircularChartDataLabel3D]} />
<CircularChart3DSeriesCollectionDirective>
  <CircularChart3DSeriesDirective
    dataSource={circularData}
    xName="x"
    yName="y"
    dataLabel={{
      visible: true,
      position: 'Outside',
      connectorStyle: { length: '30px', color: 'red', dashArray:
'2.45', width: 2 },
    }}
  ></CircularChart3DSeriesDirective>
</CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92, text: 'Chrome: 40%' },
    { x: 'Internet Explorer', y: 6.12, text: 'Chrome: 15%' },
    { x: 'Opera', y: 3.15, text: 'Chrome: 8%' },
    { x: 'Edge', y: 5.5, text: 'Chrome: 15%' },
    { x: 'Safari', y: 19.97, text: 'Chrome: 20%' },
    { x: 'Others', y: 2.34, text: 'Chrome: 2%' },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      legendSettings={{ visible: true, position: 'Right' }}
    >
      <Inject services={[PieSeries3D, CircularChartDataLabel3D]} />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"

```

```

        yName="y"
        dataLabel={{
            visible: true,
            position: 'Outside',
            connectorStyle: { length: '30px', color: 'red', dashArray:
'2.45', width: 2 },
        }}
    </CircularChart3DSeriesDirective>
</CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Text mapping

Text from the data source can be mapped using the `name` property within the data label.

INDEX.JSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartDataLabel3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData = [
        { x: 'Chrome', y: 62.92, text: 'Chrome: 40%' },
        { x: 'Internet Explorer', y: 6.12, text: 'Chrome: 15%' },
        { x: 'Opera', y: 3.15, text: 'Chrome: 8%' },
        { x: 'Edge', y: 5.5, text: 'Chrome: 15%' },
        { x: 'Safari', y: 19.97, text: 'Chrome: 20%' },
        { x: 'Others', y: 2.34, text: 'Chrome: 2%' },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Browser Market Shares in November 2023"
            tilt={-45}
            legendSettings={{ visible: true, position: 'Right' }}
        >
            <Inject services={[PieSeries3D, CircularChartDataLabel3D]} />
            <CircularChart3DSeriesCollectionDirective>
                <CircularChart3DSeriesDirective
                    dataSource={circularData}
                    xName="x"
                    yName="y"
                    dataLabel={{
                        visible: true,

```

```

        position: 'Outside',
        name: 'text'
    }}
    </CircularChart3DSeriesDirective>
</CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartDataLabel3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData: any[] = [
        { x: 'Chrome', y: 62.92, text: 'Chrome: 40%' },
        { x: 'Internet Explorer', y: 6.12, text: 'Chrome: 15%' },
        { x: 'Opera', y: 3.15, text: 'Chrome: 8%' },
        { x: 'Edge', y: 5.5, text: 'Chrome: 15%' },
        { x: 'Safari', y: 19.97, text: 'Chrome: 20%' },
        { x: 'Others', y: 2.34, text: 'Chrome: 2%' },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Browser Market Shares in November 2023"
            tilt={-45}
            legendSettings={{ visible: true, position: 'Right' }}
        >
            <Inject services={[PieSeries3D, CircularChartDataLabel3D]} />
            <CircularChart3DSeriesCollectionDirective>
                <CircularChart3DSeriesDirective
                    dataSource={circularData}
                    xName="x"
                    yName="y"
                    dataLabel={{
                        visible: true,
                        position: 'Outside',
                        name: 'text'
                    }}
                ></CircularChart3DSeriesDirective>
            </CircularChart3DSeriesCollectionDirective>
        </CircularChart3DComponent>
    );
}

```

```

}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Format

The data label for the 3D Circular Chart can be formatted using the `format` property. You can utilize global formatting options such as 'n', 'p', and 'c'.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92, text: 'Chrome: 40%' },
    { x: 'Internet Explorer', y: 6.12, text: 'Chrome: 15%' },
    { x: 'Opera', y: 3.15, text: 'Chrome: 8%' },
    { x: 'Edge', y: 5.5, text: 'Chrome: 15%' },
    { x: 'Safari', y: 19.97, text: 'Chrome: 20%' },
    { x: 'Others', y: 2.34, text: 'Chrome: 2%' },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      legendSettings={{ visible: true, position: 'Right' }}
    >
      <Inject services={[PieSeries3D, CircularChartDataLabel3D]} />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          dataLabel={{
            visible: true,
            position: 'Outside',
            format: 'c1'
          }}
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));

```

```
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92, text: 'Chrome: 40%' },
    { x: 'Internet Explorer', y: 6.12, text: 'Chrome: 15%' },
    { x: 'Opera', y: 3.15, text: 'Chrome: 8%' },
    { x: 'Edge', y: 5.5, text: 'Chrome: 15%' },
    { x: 'Safari', y: 19.97, text: 'Chrome: 20%' },
    { x: 'Others', y: 2.34, text: 'Chrome: 2%' },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      legendSettings={{ visible: true, position: 'Right' }}
    >
      <Inject services={[PieSeries3D, CircularChartDataLabel3D]} />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          dataLabel={{
            visible: true,
            position: 'Outside',
            format: 'c1'
          }}
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}
```

Value	Format	Resultant Value	Description
1000	n1	1000.0	The number is rounded to 1 decimal place.

1000	n2	1000.00	The number is rounded to 2 decimal places.
1000	n3	1000.000	The number is rounded to 3 decimal place.
0.01	p1	1.0%	The number is converted to percentage with 1 decimal place.
0.01	p2	1.00%	The number is converted to percentage with 2 decimal place.
0.01	p3	1.000%	The number is converted to percentage with 3 decimal place.
1000	c1	\$1000.0	The currency symbol is appended to number and number is rounded to 1 decimal place.
1000	c2	\$1000.00	The currency symbol is appended to number and number is rounded to 2 decimal place.

Customization

Individual text for the data points in the 3D Circular Chart can be customized using the `textRender` event.

INDEX.JSX

```
{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92, text: 'Chrome: 40%' },
    { x: 'Internet Explorer', y: 6.12, text: 'Chrome: 15%' },
    { x: 'Opera', y: 3.15, text: 'Chrome: 8%' },
    { x: 'Edge', y: 5.5, text: 'Chrome: 15%' },
    { x: 'Safari', y: 19.97, text: 'Chrome: 20%' },
    { x: 'Others', y: 2.34, text: 'Chrome: 2%' },
  ];
  const textRender = function (args) {
    if (args.point.index === 0) {
      args.text = 'Custom Datalabel';
    }
  };
  return (
    <CircularChart3DComponent
      id="charts"
      textRender={textRender}
      title="Browser Market Shares in November 2023"
      tilt={-45}
      legendSettings={{ visible: true, position: 'Right' }}
    >
```

```

    <Inject services={[PieSeries3D, CircularChartDataLabel3D]} />
    <CircularChart3DSeriesCollectionDirective>
      <CircularChart3DSeriesDirective
        dataSource={circularData}
        xName="x"
        yName="y"
        dataLabel={{
          visible: true,
          enableRotation: true,
        }}
      ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
  </CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92, text: 'Chrome: 40%' },
    { x: 'Internet Explorer', y: 6.12, text: 'Chrome: 15%' },
    { x: 'Opera', y: 3.15, text: 'Chrome: 8%' },
    { x: 'Edge', y: 5.5, text: 'Chrome: 15%' },
    { x: 'Safari', y: 19.97, text: 'Chrome: 20%' },
    { x: 'Others', y: 2.34, text: 'Chrome: 2%' },
  ];
  const textRender = function (args) {
    if (args.point.index === 0) {
      args.text = 'Custom Datalabel';
    }
  };
  return (
    <CircularChart3DComponent
      id="charts"
      textRender={textRender}
      title="Browser Market Shares in November 2023"
      tilt={-45}
    >
      <Inject services={[PieSeries3D, CircularChartDataLabel3D]} />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective

```



```

        dataSource={circularData}
        xName="x"
        yName="y"
        dataLabel={{
            visible: true,
            enableRotation: true,
        }}
    </CircularChart3DSeriesDirective>
</CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Using textRender event

You can customize the data label of a pie chart using the `textRender` event as follows to show the percentage.

INDEX.JSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartDataLabel3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData = [
        { x: 'Chrome', y: 62.92, text: 'Chrome: 40%' },
        { x: 'Internet Explorer', y: 6.12, text: 'Chrome: 15%' },
        { x: 'Opera', y: 3.15, text: 'Chrome: 8%' },
        { x: 'Edge', y: 5.5, text: 'Chrome: 15%' },
        { x: 'Safari', y: 19.97, text: 'Chrome: 20%' },
        { x: 'Others', y: 2.34, text: 'Chrome: 2%' },
    ];
    const textRender = function (args) {
        args.text = args.text + '%';
    };
    return (
        <CircularChart3DComponent
            id="charts"
            textRender={textRender}
            title="Browser Market Shares in November 2023"
            tilt={-45}
            legendSettings={{ visible: true, position: 'Right' }}
        >
            <Inject services={[PieSeries3D, CircularChartDataLabel3D]} />
            <CircularChart3DSeriesCollectionDirective>
                <CircularChart3DSeriesDirective

```

```

        dataSource={circularData}
        xName="x"
        yName="y"
        dataLabel={{
            visible: true,
            enableRotation: true,
        }}
    </CircularChart3DSeriesDirective>
</CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartDataLabel3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData: any[] = [
        { x: 'Chrome', y: 62.92, text: 'Chrome: 40%' },
        { x: 'Internet Explorer', y: 6.12, text: 'Chrome: 15%' },
        { x: 'Opera', y: 3.15, text: 'Chrome: 8%' },
        { x: 'Edge', y: 5.5, text: 'Chrome: 15%' },
        { x: 'Safari', y: 19.97, text: 'Chrome: 20%' },
        { x: 'Others', y: 2.34, text: 'Chrome: 2%' },
    ];
    const textRender = function (args) {
        args.text = args.text + '%';
    };
    return (
        <CircularChart3DComponent
            id="charts"
            textRender={textRender}
            title="Browser Market Shares in November 2023"
            tilt={-45}
            legendSettings={{ visible: true, position: 'Right' }}
        >
            <Inject services={[PieSeries3D, CircularChartDataLabel3D]} />
            <CircularChart3DSeriesCollectionDirective>
                <CircularChart3DSeriesDirective
                    dataSource={circularData}
                    xName="x"
                    yName="y"
                    dataLabel={{

```

```

        visible: true,
        enableRotation: true,
      }}
    </CircularChart3DSeriesDirective>
  </CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Using template

You can display the percentage values in the data label of a pie chart using the `template` option.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92, text: 'Chrome: 40%' },
    { x: 'Internet Explorer', y: 6.12, text: 'Chrome: 15%' },
    { x: 'Opera', y: 3.15, text: 'Chrome: 8%' },
    { x: 'Edge', y: 5.5, text: 'Chrome: 15%' },
    { x: 'Safari', y: 19.97, text: 'Chrome: 20%' },
    { x: 'Others', y: 2.34, text: 'Chrome: 2%' },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      legendSettings={{ visible: true, position: 'Right' }}
    >
      <Inject services={[PieSeries3D, CircularChartDataLabel3D]} />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          dataLabel={{
            visible: true,
            enableRotation: true,
            template: '${point.y}%'
          }}
        >

```

```

        </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% enddraw %}

```

INDEX.TSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartDataLabel3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData: any[] = [
        { x: 'Chrome', y: 62.92, text: 'Chrome: 40%' },
        { x: 'Internet Explorer', y: 6.12, text: 'Chrome: 15%' },
        { x: 'Opera', y: 3.15, text: 'Chrome: 8%' },
        { x: 'Edge', y: 5.5, text: 'Chrome: 15%' },
        { x: 'Safari', y: 19.97, text: 'Chrome: 20%' },
        { x: 'Others', y: 2.34, text: 'Chrome: 2%' },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Browser Market Shares in November 2023"
            tilt={-45}
            legendSettings={{ visible: true, position: 'Right' }}
        >
            <Inject services={[PieSeries3D, CircularChartDataLabel3D]} />
            <CircularChart3DSeriesCollectionDirective>
                <CircularChart3DSeriesDirective
                    dataSource={circularData}
                    xName="x"
                    yName="y"
                    dataLabel={{
                        visible: true,
                        enableRotation: true,
                        template: '${point.y}%'
                    }}
                ></CircularChart3DSeriesDirective>
            </CircularChart3DSeriesCollectionDirective>
        </CircularChart3DComponent>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Empty points in React 3D Circular Chart component

Data points containing `null` or `undefined` values are considered empty points. These empty data points are ignored and not plotted in the 3D Circular Chart. You can customize the handling of empty points using the `emptyPointSettings` property in the series. The default mode for empty points is `Gap`. Other supported modes include `Average`, `Drop`, and `Zero`.

INDEX.JSX

```
{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Jan', y: 3 }, { x: 'Feb', y: 3.5 },
    { x: 'Mar', y: undefined }, { x: 'Apr', y: 13.5 },
    { x: 'May', y: 19 }, { x: 'Jun', y: 23.5 },
    { x: 'Jul', y: null }, { x: 'Aug', y: 25 },
    { x: 'Sep', y: 21 }, { x: 'Oct', y: 15 }
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          dataLabel={{
            visible: true,
          }} emptyPointSettings={{ mode: 'Zero' }}
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
```

```
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Jan', y: 3 }, { x: 'Feb', y: 3.5 },
    { x: 'Mar', y: undefined }, { x: 'Apr', y: 13.5 },
    { x: 'May', y: 19 }, { x: 'Jun', y: 23.5 },
    { x: 'Jul', y: null }, { x: 'Aug', y: 25 },
    { x: 'Sep', y: 21 }, { x: 'Oct', y: 15 }
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          dataLabel={{
            visible: true,
          }} emptyPointSettings={{ mode: 'Zero' }}
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}
```

Customization

A specific color for an empty point can be set by using the `fill` property in `emptyPointSettings`.

INDEX.JSX

```
{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Jan', y: 3 }, { x: 'Feb', y: 3.5 },
    { x: 'Mar', y: undefined }, { x: 'Apr', y: 13.5 },
    { x: 'May', y: 19 }, { x: 'Jun', y: 23.5 },
    { x: 'Jul', y: null }, { x: 'Aug', y: 25 },
    { x: 'Sep', y: 21 }, { x: 'Oct', y: 15 }
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          dataLabel={{
            visible: true,
          }} emptyPointSettings={{ mode: 'Average', fill: 'yellow' }}
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
```

```

CircularChart3DSeriesDirective,
PieSeries3D,
CircularChartDataLabel3D,
Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Jan', y: 3 }, { x: 'Feb', y: 3.5 },
    { x: 'Mar', y: undefined }, { x: 'Apr', y: 13.5 },
    { x: 'May', y: 19 }, { x: 'Jun', y: 23.5 },
    { x: 'Jul', y: null }, { x: 'Aug', y: 25 },
    { x: 'Sep', y: 21 }, { x: 'Oct', y: 15 }
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          dataLabel={{
            visible: true,
          }} emptyPointSettings={{ mode: 'Average', fill: 'yellow' }}
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Legend in React 3D Circular Chart component

The legend provides information about the data points rendered in the 3D Circular Chart. It can be added by enabling the `visible` option in the `legendSettings` property.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,

```



```

CircularChartLegend3D,
Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Legend in Circular 3D"
      tilt={-45}
      legendSettings={{ visible: true }}
    >
      <Inject services={[PieSeries3D, CircularChartLegend3D]} />
      <CircularChart3DSeriesCollectionDirective
        >
          <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
          ></CircularChart3DSeriesDirective>
        </CircularChart3DSeriesCollectionDirective>
      </CircularChart3DComponent>
    );
  }
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartLegend3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },

```

```

    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Legend in Circular 3D"
      tilt={-45}
      legendSettings={{ visible: true }}
    >
      <Inject services={[PieSeries3D, CircularChartLegend3D]} />
      <CircularChart3DSeriesCollectionDirective
        >
          <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
          ></CircularChart3DSeriesDirective>
        </CircularChart3DSeriesCollectionDirective>
      </CircularChart3DComponent>
    );
  }
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

To use data label feature, we need to inject `CircularChartLegend3D` module into the services.

Position and alignment

By using the `position` property, the legend can be positioned at the `left`, `right`, `top` or `bottom` of the 3D Circular Chart. By default, the legend will be positioned to the right of the 3D Circular Chart.

Additionally, you can align the legend to the `center`, `far` or `near` of the chart using the `alignment` property.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartLegend3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];

```

```

];
return (
  <CircularChart3DComponent
    id="charts"
    title="Legend in Circular 3D"
    tilt={-45}
    legendSettings={{ visible: true, position: 'Right' }}
  >
    <Inject services={[PieSeries3D, CircularChartLegend3D]} />
    <CircularChart3DSeriesCollectionDirective
      >
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% enddraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartLegend3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Legend in Circular 3D"
      tilt={-45}
      legendSettings={{ visible: true, position: 'Right' }}
    >
      <Inject services={[PieSeries3D, CircularChartLegend3D]} />
      <CircularChart3DSeriesCollectionDirective
        >

```

```

        <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
        ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Legend reverse

You can reverse the order of the legend items by using the `reverse` property in `legendSettings`. By default, the legend for the first series in the collection will be placed first.

INDEX.JSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartLegend3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData = [
        { x: 'Chrome', y: 62.92 },
        { x: 'Internet Explorer', y: 6.12 },
        { x: 'Opera', y: 3.15 },
        { x: 'Edge', y: 5.5 },
        { x: 'Safari', y: 19.97 },
        { x: 'Others', y: 2.34 },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Legend in Circular 3D"
            tilt={-45}
            legendSettings={{ visible: true, position: 'Right', reverse: true }}
        >
            <Inject services={[PieSeries3D, CircularChartLegend3D]} />
            <CircularChart3DSeriesCollectionDirective
                >
                    <CircularChart3DSeriesDirective
                        dataSource={circularData}
                        xName="x"
                        yName="y"
                    ></CircularChart3DSeriesDirective>
                </CircularChart3DSeriesCollectionDirective>
            </CircularChart3DComponent>

```

```

    );
  }
  export default App;
  ReactDOM.render(<App />, document.getElementById('charts'));
  {% enddraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartLegend3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Legend in Circular 3D"
      tilt={-45}
      legendSettings={{ visible: true, position: 'Right', reverse: true }}
    >
      <Inject services={[PieSeries3D, CircularChartLegend3D]} />
      <CircularChart3DSeriesCollectionDirective
        >
          <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
          ></CircularChart3DSeriesDirective>
        </CircularChart3DSeriesCollectionDirective>
      </CircularChart3DComponent>
    );
  }
  export default App;
  ReactDOM.render(<App />, document.getElementById('charts'));
  {% enddraw %}

```

Legend shape

To change the legend shape, use the `legendShape` property in the `series`. By default, the legend shape is set to `seriesType`.

INDEX.JSX

```
{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartLegend3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Legend in Circular 3D"
      tilt={-45}
      legendSettings={{ visible: true }}
    >
      <Inject services={[PieSeries3D, CircularChartLegend3D]} />
      <CircularChart3DSeriesCollectionDirective
        >
          <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            legendShape='Rectangle'
          ></CircularChart3DSeriesDirective>
        </CircularChart3DSeriesCollectionDirective>
      </CircularChart3DComponent>
    );
  }
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartLegend3D,
  Inject,
}
```

```

} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Legend in Circular 3D"
      tilt={-45}
      legendSettings={{ visible: true }}
    >
      <Inject services={[PieSeries3D, CircularChartLegend3D]} />
      <CircularChart3DSeriesCollectionDirective
        >
          <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            legendShape='Rectangle'
          ></CircularChart3DSeriesDirective>
        </CircularChart3DSeriesCollectionDirective>
      </CircularChart3DComponent>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Legend size

The legend size can be changed by using the `width` and `height` properties in `legendSettings`.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartLegend3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },

```

```

    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Legend in Circular 3D"
      tilt={-45}
      legendSettings={{ visible: true, width: '200px', height: '100px' }}
    >
      <Inject services={[PieSeries3D, CircularChartLegend3D]} />
      <CircularChart3DSeriesCollectionDirective
        >
          <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            legendShape='Rectangle'
          ></CircularChart3DSeriesDirective>
        </CircularChart3DSeriesCollectionDirective>
      </CircularChart3DComponent>
    );
  }
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% enddraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartLegend3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Legend in Circular 3D"
      tilt={-45}
    >

```



```

    legendSettings={{ visible: true, width: '200px', height: '100px' }}
  >
  <Inject services={[PieSeries3D, CircularChartLegend3D]} />
  <CircularChart3DSeriesCollectionDirective
  >
    <CircularChart3DSeriesDirective
      dataSource={circularData}
      xName="x"
      yName="y"
      legendShape='Rectangle'
    ></CircularChart3DSeriesDirective>
  </CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Legend item size

The size of the legend items can be customized by using the `shapeHeight` and `shapeWidth` properties in `legendSettings`.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartLegend3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Legend in Circular 3D"
      tilt={-45}
      legendSettings={{ visible: true, shapeHeight: 2, shapeWidth: 2 }}
    >
      <Inject services={[PieSeries3D, CircularChartLegend3D]} />
      <CircularChart3DSeriesCollectionDirective
      >
        <CircularChart3DSeriesDirective

```

```

        dataSource={circularData}
        xName="x"
        yName="y"
        legendShape='Rectangle'
    <</CircularChart3DSeriesDirective>
  </CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartLegend3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Legend in Circular 3D"
      tilt={-45}
      legendSettings={{ visible: true, shapeHeight: 2, shapeWidth: 2 }}
    >
      <Inject services={[PieSeries3D, CircularChartLegend3D]} />
      <CircularChart3DSeriesCollectionDirective
        >
          <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            legendShape='Rectangle'
          ></CircularChart3DSeriesDirective>
        </CircularChart3DSeriesCollectionDirective>
      </CircularChart3DComponent>
    );
  }
export default App;

```

```
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}
```

Legend paging

Paging will be enabled by default when the legend items exceed the legend bounds. Each legend item can be viewed by navigating between the pages using navigation buttons.

INDEX.JSX

```
{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartLegend3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Legend in Circular 3D"
      tilt={-45}
      legendSettings={{ visible: true, width: '200px', height: '100px' }}
    >
      <Inject services={[PieSeries3D, CircularChartLegend3D]} />
      <CircularChart3DSeriesCollectionDirective
        >
          <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            legendShape='Rectangle'
          ></CircularChart3DSeriesDirective>
        </CircularChart3DSeriesCollectionDirective>
      </CircularChart3DComponent>
    >
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartLegend3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Legend in Circular 3D"
      tilt={-45}
      legendSettings={{ visible: true, width: '200px', height: '100px' }}
    >
      <Inject services={[PieSeries3D, CircularChartLegend3D]} />
      <CircularChart3DSeriesCollectionDirective
        >
          <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            legendShape='Rectangle'
          ></CircularChart3DSeriesDirective>
        </CircularChart3DSeriesCollectionDirective>
      </CircularChart3DComponent>
    );
  }
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}
```

Legend text wrap

When the legend text exceeds the container, the text can be wrapped using the `textWrap` property in `legendSettings`. End users can also wrap the legend text based on the `maximumLabelWidth` property in `legendSettings`.

INDEX.JSX

```
{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
```

```

CircularChart3DSeriesDirective,
PieSeries3D,
CircularChartLegend3D,
Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData = [
        { x: 'Chrome', y: 62.92 },
        { x: 'Internet Explorer', y: 6.12 },
        { x: 'Opera', y: 3.15 },
        { x: 'Edge', y: 5.5 },
        { x: 'Safari', y: 19.97 },
        { x: 'Others', y: 2.34 },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Legend in Circular 3D"
            tilt={-45}
            legendSettings={{ visible: true, maximumLabelWidth: 80, textWrap:
'Wrap' }}
        >
            <Inject services={[PieSeries3D, CircularChartLegend3D]} />
            <CircularChart3DSeriesCollectionDirective
                >
                    <CircularChart3DSeriesDirective
                        dataSource={circularData}
                        xName="x"
                        yName="y"
                        legendShape='Rectangle'
                    ></CircularChart3DSeriesDirective>
                </CircularChart3DSeriesCollectionDirective>
            </CircularChart3DComponent>
        </CircularChart3DComponent>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartLegend3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData: any[] = [

```

```

    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Legend in Circular 3D"
      tilt={-45}
      legendSettings={{ visible: true, maximumLabelWidth: 80, textWrap:
'Wrap' }}
    >
      <Inject services={[PieSeries3D, CircularChartLegend3D]} />
      <CircularChart3DSeriesCollectionDirective
        >
          <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            legendShape='Rectangle'
          ></CircularChart3DSeriesDirective>
        </CircularChart3DSeriesCollectionDirective>
      </CircularChart3DComponent>
    );
  }
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Legend title

You can set a title for the legend using the `title` property in `legendSettings`. The `size`, `color`, `opacity`, `fontStyle`, `fontWeight`, `fontFamily`, `textAlignment`, and `textOverflow` of the legend title can be customized using the `titleStyle` property in `legendSettings`.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartLegend3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },

```

```

    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Legend in Circular 3D"
      tilt={-45}
      legendSettings={{ title: 'Legend Title', visible: true }}
    >
      <Inject services={[PieSeries3D, CircularChartLegend3D]} />
      <CircularChart3DSeriesCollectionDirective
        >
          <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            legendShape='Rectangle'
          ></CircularChart3DSeriesDirective>
        </CircularChart3DSeriesCollectionDirective>
      </CircularChart3DComponent>
    );
  }
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartLegend3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Legend in Circular 3D"
      tilt={-45}
      legendSettings={{ title: 'Legend Title', visible: true }}
    >

```

```

    >
    <Inject services={[PieSeries3D, CircularChartLegend3D]} />
    <CircularChart3DSeriesCollectionDirective
    >
        <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            legendShape='Rectangle'
        ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Arrow page navigation

The page number will always be visible when using legend paging. However, it is now possible to disable the page number and enable page navigation with the left and right arrows. To render the arrow page navigation, set the `enablePages` property to **false**.

INDEX.JSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartLegend3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData = [
        { x: 'Chrome', y: 62.92 },
        { x: 'Internet Explorer', y: 6.12 },
        { x: 'Opera', y: 3.15 },
        { x: 'Edge', y: 5.5 },
        { x: 'Safari', y: 19.97 },
        { x: 'Others', y: 2.34 },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Legend in Circular 3D"
            tilt={-45}
            legendSettings={{
                visible: true,
                width: '240px',
                height: '100px',
                enablePages: false,
            }}
        >

```



```

    >
    <Inject services={[PieSeries3D, CircularChartLegend3D]} />
    <CircularChart3DSeriesCollectionDirective
    >
      <CircularChart3DSeriesDirective
        dataSource={circularData}
        xName="x"
        yName="y"
        legendShape='Rectangle'
      ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
  </CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartLegend3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Legend in Circular 3D"
      tilt={-45}
      legendSettings={{
        visible: true,
        width: '240px',
        height: '100px',
        enablePages: false,
      }}
    >
      <Inject services={[PieSeries3D, CircularChartLegend3D]} />
      <CircularChart3DSeriesCollectionDirective
      >
        <CircularChart3DSeriesDirective

```

```

        dataSource={circularData}
        xName="x"
        yName="y"
        legendShape='Rectangle'
    ></CircularChart3DSeriesDirective>
</CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Legend item padding

The `itemPadding` property can be used to adjust the space between the legend items.

INDEX.JSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartLegend3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData = [
        { x: 'Chrome', y: 62.92 },
        { x: 'Internet Explorer', y: 6.12 },
        { x: 'Opera', y: 3.15 },
        { x: 'Edge', y: 5.5 },
        { x: 'Safari', y: 19.97 },
        { x: 'Others', y: 2.34 },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Legend in Circular 3D"
            tilt={-45}
            legendSettings={{
                visible: true,
                position: 'Bottom',
                itemPadding: 40
            }}
        >
            <Inject services={[PieSeries3D, CircularChartLegend3D]} />
            <CircularChart3DSeriesCollectionDirective
                >
                    <CircularChart3DSeriesDirective
                        dataSource={circularData}
                        xName="x"
                        yName="y"

```

```

        legendShape='Rectangle'
      ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
  </CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartLegend3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Legend in Circular 3D"
      tilt={-45}
      legendSettings={{
        visible: true,
        position: 'Bottom',
        itemPadding: 40
      }}
    >
      <Inject services={[PieSeries3D, CircularChartLegend3D]} />
      <CircularChart3DSeriesCollectionDirective
        >
          <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            legendShape='Rectangle'
          ></CircularChart3DSeriesDirective>
        </CircularChart3DSeriesCollectionDirective>
      </CircularChart3DComponent>
    );
}

```

```
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}
```

Tooltip in React 3D Circular Chart component

The 3D Circular Chart will display details about the points through a tooltip, when the mouse is moved over a specific point. By default, the tooltip is not visible. It can be enabled by using the `enable` property in `tooltip` to `true`.

INDEX.JSX

```
{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  CircularChartTooltip3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      tooltip={{enable: true}}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
          CircularChartTooltip3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          dataLabel={{
            visible: true,
          }}
        ></CircularChart3DSeriesDirective>
```

```

        </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartDataLabel3D,
    CircularChartTooltip3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData: any[] = [
        { x: 'Chrome', y: 62.92 },
        { x: 'Internet Explorer', y: 6.12 },
        { x: 'Opera', y: 3.15 },
        { x: 'Edge', y: 5.5 },
        { x: 'Safari', y: 19.97 },
        { x: 'Others', y: 2.34 },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Browser Market Shares in November 2023"
            tilt={-45}
            tooltip={{enable: true}}
        >
            <Inject
                services={[
                    PieSeries3D,
                    CircularChartDataLabel3D,
                    CircularChartTooltip3D,
                ]}
            />
            <CircularChart3DSeriesCollectionDirective>
                <CircularChart3DSeriesDirective
                    dataSource={circularData}
                    xName="x"
                    yName="y"
                    dataLabel={{
                        visible: true,
                    }}
                ></CircularChart3DSeriesDirective>
            </CircularChart3DSeriesCollectionDirective>
        </CircularChart3DComponent>
    );
}
ReactDOM.render(<App />, document.getElementById('charts'));

```

```

    );
  }
  export default App;
  ReactDOM.render(<App />, document.getElementById('charts'));
  {% enddraw %}

```

To use data label feature, we need to inject `CircularChartTooltip3D` module into the services.

Header

You can specify a header for the tooltip by using the `header` property in `tooltip`.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  CircularChartTooltip3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      tooltip={{enable: true, header: 'Browser'}}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
          CircularChartTooltip3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          dataLabel={{
            visible: true,
          }}
        >

```

```

        </CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% enddraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  CircularChartTooltip3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      tooltip={{enable: true, header: 'Browser'}}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
          CircularChartTooltip3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          dataLabel={{
            visible: true,
          }}
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
ReactDOM.render(<App />, document.getElementById('charts'));

```

```

    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Format

By default, the tooltip shows information about the x and y values in points. Additionally, more information can be displayed in the tooltip by using the `format` property. For example, the format `${series.name} : ${point.x}` shows the series name and the point's x value.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  CircularChartTooltip3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      tooltip={{ enable: true, format: '${point.x}' }}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
          CircularChartTooltip3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          dataLabel={{
            visible: true,

```



```

    }}
    </CircularChart3DSeriesDirective>
  </CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  CircularChartTooltip3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      tooltip={{ enable: true, format: '${point.x}' }}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
          CircularChartTooltip3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          dataLabel={{
            visible: true,
          }}
        ></CircularChart3DSeriesDirective>

```

```

    </CircularChart3DSeriesCollectionDirective>
  </CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Tooltip template

Any HTML elements can be displayed in the tooltip by using the `template` property in the tooltip.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  CircularChartTooltip3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      tooltip={{ enable: true, template: '<div> ${x} : ${y} <div>' }}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
          CircularChartTooltip3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          dataLabel={{
            visible: true,
          }}
        >

```

```

        </CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% enddraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  CircularChartTooltip3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      tooltip={{ enable: true, template: '<div> ${x} : ${y} <div>' }}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
          CircularChartTooltip3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          dataLabel={{
            visible: true,
          }}
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

```

    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Fixed tooltip

By default, the tooltip tracks the mouse movement, but it can be set to a fixed position using the `location` property in `tooltip`.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  CircularChartTooltip3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      tooltip={{ enable: true, location: {x: 200, y: 90} }}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
          CircularChartTooltip3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          dataLabel={{
            visible: true,
          }}

```

```

        </CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% enddraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  CircularChartTooltip3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      tooltip={{ enable: true, location: {x: 200, y: 90} }}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
          CircularChartTooltip3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"
          yName="y"
          dataLabel={{
            visible: true,
          }}
        ></CircularChart3DSeriesDirective>
      </CircularChart3DSeriesCollectionDirective>
    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

```

    </CircularChart3DComponent>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Customization

The **fill** and **border** properties are used to customize the background color and border of the tooltip, respectively. The **textStyle** property in the tooltip is used to customize the font of the tooltip text. Additionally, the **highlightColor** property can be used to change the color of the data point when hovering.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  CircularChartTooltip3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      tooltip={{ enable: true, fill: 'red', border:{width: 1.45, color :
'white'}} }}
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
          CircularChartTooltip3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}
          xName="x"

```

```

        yName="y"
        dataLabel={{
            visible: true,
        }}
    </CircularChart3DSeriesDirective>
</CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartDataLabel3D,
    CircularChartTooltip3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData: any[] = [
        { x: 'Chrome', y: 62.92 },
        { x: 'Internet Explorer', y: 6.12 },
        { x: 'Opera', y: 3.15 },
        { x: 'Edge', y: 5.5 },
        { x: 'Safari', y: 19.97 },
        { x: 'Others', y: 2.34 },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Browser Market Shares in November 2023"
            tilt={-45}
            tooltip={{ enable: true, fill: 'red', border:{width: 1.45, color :
'white'}} }}
        >
            <Inject
                services={[
                    PieSeries3D,
                    CircularChartDataLabel3D,
                    CircularChartTooltip3D,
                ]}
            />
            <CircularChart3DSeriesCollectionDirective>
                <CircularChart3DSeriesDirective
                    dataSource={circularData}
                    xName="x"
                    yName="y"

```

```

        dataLabel={{
            visible: true,
        }}
    </CircularChart3DSeriesDirective>
</CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Customization of individual tooltip

Using the `tooltipRender` event, you can customize tooltip values for a particular point.

INDEX.JSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartDataLabel3D,
    CircularChartTooltip3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData = [
        { x: 'Chrome', y: 62.92 },
        { x: 'Internet Explorer', y: 6.12 },
        { x: 'Opera', y: 3.15 },
        { x: 'Edge', y: 5.5 },
        { x: 'Safari', y: 19.97 },
        { x: 'Others', y: 2.34 },
    ];
    const textRender = function (args) {
        if (args.point.index === 0) {
            args.text = 'Custom Tooltip';
        }
    };
    return (
        <CircularChart3DComponent
            id="charts"
            title="Browser Market Shares in November 2023"
            tilt={-45}
            tooltipRender={textRender}
            tooltip={{enable:true}}
        >
            <Inject
                services={[
                    PieSeries3D,
                    CircularChartDataLabel3D,
                    CircularChartTooltip3D,

```



```

    ]}
  />
  <CircularChart3DSeriesCollectionDirective>
    <CircularChart3DSeriesDirective
      dataSource={circularData}
      xName="x"
      yName="y"
      dataLabel={{
        visible: true,
      }}
    ></CircularChart3DSeriesDirective>
  </CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  CircularChartTooltip3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  const textRender = function (args) {
    if (args.point.index === 0) {
      args.text = 'Custom Tooltip';
    }
  };
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      tooltipRender={textRender}
      tooltip={{ enable: true }}
    >
      <Inject

```

```

        services={ [
            PieSeries3D,
            CircularChartDataLabel3D,
            CircularChartTooltip3D,
        ] }
    />
    <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            dataLabel={{
                visible: true,
            }}
        ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Title and subtitle in React 3D Circular Chart component

Title

The 3D Circular Chart can be given a title by using the `title` property to display information about the plotted data.

INDEX.JSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartDataLabel3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData = [
        { x: 'Chrome', y: 62.92 },
        { x: 'Internet Explorer', y: 6.12 },
        { x: 'Opera', y: 3.15 },
        { x: 'Edge', y: 5.5 },
        { x: 'Safari', y: 19.97 },
        { x: 'Others', y: 2.34 },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Browser Market Shares in November 2023"
            tilt={-45}

```

```

    legendSettings={{ visible: true, position: 'Right' }}
  >
  <Inject
    services={[
      PieSeries3D,
      CircularChartDataLabel3D,
    ]}
  />
  <CircularChart3DSeriesCollectionDirective>
    <CircularChart3DSeriesDirective
      dataSource={circularData}
      xName="x"
      yName="y"
      dataLabel={{
        visible: true
      }}
    ></CircularChart3DSeriesDirective>
  </CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      legendSettings={{ visible: true, position: 'Right' }}
    >
      <Inject
        services={[

```

```

        PieSeries3D,
        CircularChartDataLabel3D,
    ]}
    />
    <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            dataLabel={{
                visible: true
            }}
        ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Title customization

The title of the 3D Circular Chart can be customized using the `titleStyle` property.

INDEX.JSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartDataLabel3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData = [
        { x: 'Chrome', y: 62.92 },
        { x: 'Internet Explorer', y: 6.12 },
        { x: 'Opera', y: 3.15 },
        { x: 'Edge', y: 5.5 },
        { x: 'Safari', y: 19.97 },
        { x: 'Others', y: 2.34 },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Browser Market Shares in November 2023"
            tilt={-45}
            titleStyle={{fontFamily: 'Cursive', color: 'Blue', textAlign:
'Far'}}
        >
            <Inject
                services=[

```

```

        PieSeries3D,
        CircularChartDataLabel3D,
    ]}
    />
    <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
            dataSource={circularData}
            xName="x"
            yName="y"
            dataLabel={{
                visible: true
            }}
        ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartDataLabel3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData: any[] = [
        { x: 'Chrome', y: 62.92 },
        { x: 'Internet Explorer', y: 6.12 },
        { x: 'Opera', y: 3.15 },
        { x: 'Edge', y: 5.5 },
        { x: 'Safari', y: 19.97 },
        { x: 'Others', y: 2.34 },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Browser Market Shares in November 2023"
            tilt={-45}
            titleStyle= {{fontFamily: 'Cursive', color: 'Blue', textAlignment:
'Far'}}
        >
            <Inject
                services={[
                    PieSeries3D,
                    CircularChartDataLabel3D,
                ]}
            >

```

```

    />
    <CircularChart3DSeriesCollectionDirective>
      <CircularChart3DSeriesDirective
        dataSource={circularData}
        xName="x"
        yName="y"
        dataLabel={{
          visible: true
        }}
      ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
  </CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Subtitle

The 3D Circular Chart can be given a subtitle by using the `subTitle` property to display information about the plotted data.

INDEX.JSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      subTitle='Sub-title'
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
        ]}
      >

```

```

    />
    <CircularChart3DSeriesCollectionDirective>
      <CircularChart3DSeriesDirective
        dataSource={circularData}
        xName="x"
        yName="y"
        dataLabel={{
          visible: true
        }}
      ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
  </CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
  CircularChart3DComponent,
  CircularChart3DSeriesCollectionDirective,
  CircularChart3DSeriesDirective,
  PieSeries3D,
  CircularChartDataLabel3D,
  Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  const circularData: any[] = [
    { x: 'Chrome', y: 62.92 },
    { x: 'Internet Explorer', y: 6.12 },
    { x: 'Opera', y: 3.15 },
    { x: 'Edge', y: 5.5 },
    { x: 'Safari', y: 19.97 },
    { x: 'Others', y: 2.34 },
  ];
  return (
    <CircularChart3DComponent
      id="charts"
      title="Browser Market Shares in November 2023"
      tilt={-45}
      subTitle='Sub-title'
    >
      <Inject
        services={[
          PieSeries3D,
          CircularChartDataLabel3D,
        ]}
      />
      <CircularChart3DSeriesCollectionDirective>
        <CircularChart3DSeriesDirective
          dataSource={circularData}

```

```

        xName="x"
        yName="y"
        dataLabel={{
            visible: true
        }}
    ></CircularChart3DSeriesDirective>
</CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Subtitle customization

The subtitle of the 3D Circular Chart can be customized using the `subTitleStyle` property.

INDEX.JSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartDataLabel3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData = [
        { x: 'Chrome', y: 62.92 },
        { x: 'Internet Explorer', y: 6.12 },
        { x: 'Opera', y: 3.15 },
        { x: 'Edge', y: 5.5 },
        { x: 'Safari', y: 19.97 },
        { x: 'Others', y: 2.34 },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Browser Market Shares in November 2023"
            tilt={-45}
            subTitle='Sub-title'
            subTitleStyle={{fontFamily:'Cursive', fontStyle: 'Italic'}}
        >
            <Inject
                services={[
                    PieSeries3D,
                    CircularChartDataLabel3D,
                ]}
            />
            <CircularChart3DSeriesCollectionDirective>
                <CircularChart3DSeriesDirective
                    dataSource={circularData}

```



```

        xName="x"
        yName="y"
        dataLabel={{
            visible: true
        }}
    ></CircularChart3DSeriesDirective>
</CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import {
    CircularChart3DComponent,
    CircularChart3DSeriesCollectionDirective,
    CircularChart3DSeriesDirective,
    PieSeries3D,
    CircularChartDataLabel3D,
    Inject,
} from '@syncfusion/ej2-react-charts';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const circularData: any[] = [
        { x: 'Chrome', y: 62.92 },
        { x: 'Internet Explorer', y: 6.12 },
        { x: 'Opera', y: 3.15 },
        { x: 'Edge', y: 5.5 },
        { x: 'Safari', y: 19.97 },
        { x: 'Others', y: 2.34 },
    ];
    return (
        <CircularChart3DComponent
            id="charts"
            title="Browser Market Shares in November 2023"
            tilt={-45}
            subTitle='Sub-title'
            subTitleStyle={{fontFamily:'Cursive', fontStyle: 'Italic'}}
        >
            <Inject
                services={[
                    PieSeries3D,
                    CircularChartDataLabel3D,
                ]}
            />
            <CircularChart3DSeriesCollectionDirective>
                <CircularChart3DSeriesDirective
                    dataSource={circularData}
                    xName="x"
                    yName="y"
                    dataLabel={{

```

```

        visible: true
      }}
    </CircularChart3DSeriesDirective>
  </CircularChart3DSeriesCollectionDirective>
</CircularChart3DComponent>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('charts'));
{% endraw %}

```

Print and Export in React 3D Circular Chart component

Print

The rendered 3D Circular Chart can be printed directly from the browser by calling the public method **print**. The ID of the 3D Circular Chart div element must be passed as the input parameter to that method.

INDEX.JSX

```

{% raw %}
import { createRoot } from 'react-dom/client';
/**
 * Sample for Chart print
 */
import * as React from 'react';
import { useEffect, useRef } from 'react';
import {
  CircularChart3DSeriesDirective,
  CircularChart3DSeriesCollectionDirective,
  Inject,
  CircularChart3DComponent,
  CircularChartDataLabel3D,
  CircularChartLegend3D,
  PieSeries3D,
} from '@syncfusion/ej2-react-charts';
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
export let circularData = [
  { x: 'John', y: 10, dataLabelMappingName: '$10k' },
  { x: 'Jake', y: 12, dataLabelMappingName: '$12k' },
  { x: 'Peter', y: 18, dataLabelMappingName: '$18k' },
  { x: 'James', y: 11, dataLabelMappingName: '$11k' },
  { x: 'Mary', y: 9.7, dataLabelMappingName: '$9.7k' },
];
const Print = () => {
  useEffect(() => {
    const button = document.getElementById('chart-print');
    button.addEventListener('click', onClick);
  }, []);
  let chartInstance = useRef(null);
  const onClick = (e) => {
    chartInstance.current.print();
  };
  return (
    <div className="control-pane">
      <div className="control-section row">

```

```

<div className="col-lg-9">
  <CircularChart3DComponent
    id="charts"
    ref={chartInstance}
    title="Browser Market Shares in November 2023"
    tilt={-45}
    legendSettings={{ visible: false, position: 'Right' }}
  >
    <Inject
      services={[
        PieSeries3D,
        CircularChartDataLabel3D,
        CircularChartLegend3D,
      ]}
    />
    <CircularChart3DSeriesCollectionDirective>
      <CircularChart3DSeriesDirective
        dataSource={circularData}
        xName="x"
        yName="y"
        dataLabel={{
          visible: true,
          position: 'Inside',
          name: 'x',
          font: { fontWeight: '600', color: 'white' },
        }}
      ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
  </CircularChart3DComponent>
</div>
<div className="col-lg-3 property-section">
  <ButtonComponent
    id="chart-print"
    iconCss="e-icons e-print-icon"
    cssClass="e-flat"
    isPrimary={true}
  >
    Print
  </ButtonComponent>
</div>
</div>
</div>
);
};
export default Print;
createRoot(document.getElementById('charts')).render(<Print />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import { createRoot } from 'react-dom/client';
/**
 * Sample for Chart print
 */
import * as React from 'react';

```

```

import { useEffect, useRef } from 'react';
import {
  CircularChart3DSeriesDirective,
  CircularChart3DSeriesCollectionDirective,
  Inject,
  CircularChart3DComponent,
  CircularChartDataLabel3D,
  CircularChartLegend3D,
  PieSeries3D,
} from '@syncfusion/ej2-react-charts';
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
export let circularData: any[] = [
  { x: 'John', y: 10, dataLabelMappingName: '$10k' },
  { x: 'Jake', y: 12, dataLabelMappingName: '$12k' },
  { x: 'Peter', y: 18, dataLabelMappingName: '$18k' },
  { x: 'James', y: 11, dataLabelMappingName: '$11k' },
  { x: 'Mary', y: 9.7, dataLabelMappingName: '$9.7k' },
];
const Print = () => {
  useEffect(() => {
    const button = document.getElementById('chart-print');
    button.addEventListener('click', onClick);
  }, []);
  let chartInstance = useRef(null);
  const onClick = (e) => {
    chartInstance.current.print();
  };
  return (
    <div className="control-pane">
      <div className="control-section row">
        <div className="col-lg-9">
          <CircularChart3DComponent
            id="charts"
            ref={chartInstance}
            title="Browser Market Shares in November 2023"
            tilt={-45}
            legendSettings={{ visible: false, position: 'Right' }}
          >
            <Inject
              services={[
                PieSeries3D,
                CircularChartDataLabel3D,
                CircularChartLegend3D,
              ]}
            />
            <CircularChart3DSeriesCollectionDirective>
              <CircularChart3DSeriesDirective
                dataSource={circularData}
                xName="x"
                yName="y"
                dataLabel={{
                  visible: true,
                  position: 'Inside',
                  name: 'x',
                  font: { fontWeight: '600', color: 'white' },
                }}
              ></CircularChart3DSeriesDirective>
            </CircularChart3DSeriesCollectionDirective>
          </div>
        </div>
      </div>
    </div>
  );
};

```

```

        </CircularChart3DSeriesCollectionDirective>
      </CircularChart3DComponent>
    </div>
    <div className="col-lg-3 property-section">
      <ButtonComponent
        id="chart-print"
        iconCss="e-icons e-print-icon"
        cssClass="e-flat"
        isPrimary={true}
      >
        Print
      </ButtonComponent>
    </div>
  </div>
</div>
);
};
export default Print;
createRoot(document.getElementById('charts')).render(<Print />);
{% endraw %}

```

Export

The rendered 3D Circular Chart can be exported to JPEG, PNG, or SVG format using the `export` method. Additionally, you can export the 3D Circular Chart as a PDF format using the `pdfExport` method. The input parameters for this method are `type` for the format and `fileName` for the result.

INDEX.JSX

```

{% raw %}
import { createRoot } from 'react-dom/client';
/**
 * Sample for Chart print
 */
import * as React from 'react';
import { useEffect, useRef } from 'react';
import {
  CircularChart3DSeriesDirective,
  CircularChart3DSeriesCollectionDirective,
  Inject,
  CircularChart3DComponent,
  CircularChartDataLabel3D,
  CircularChartLegend3D,
  PieSeries3D,
  CircularChartExport3D,
} from '@syncfusion/ej2-react-charts';
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
export let circularData = [
  { x: 'John', y: 10, dataLabelMappingName: '$10k' },
  { x: 'Jake', y: 12, dataLabelMappingName: '$12k' },
  { x: 'Peter', y: 18, dataLabelMappingName: '$18k' },
  { x: 'James', y: 11, dataLabelMappingName: '$11k' },
  { x: 'Mary', y: 9.7, dataLabelMappingName: '$9.7k' },
];
const Print = () => {
  useEffect(() => {

```

```

    const button = document.getElementById('chart-print');
    button.addEventListener('click', onClick);
  }, []);
  let chartInstance = useRef(null);
  const onClick = (e) => {
    chartInstance.current.export('JPEG', 'chart');
  };
  return (
    <div className="control-pane">
      <div className="control-section row">
        <div className="col-lg-9">
          <CircularChart3DComponent
            id="charts"
            ref={chartInstance}
            title="Browser Market Shares in November 2023"
            tilt={-45}
            enableExport={true}
            legendSettings={{ visible: false, position: 'Right' }}
          >
            <Inject
              services={[
                PieSeries3D,
                CircularChartDataLabel3D,
                CircularChartLegend3D,
                CircularChartExport3D,
              ]}
            />
            <CircularChart3DSeriesCollectionDirective>
              <CircularChart3DSeriesDirective
                dataSource={circularData}
                xName="x"
                yName="y"
                dataLabel={{
                  visible: true,
                  position: 'Inside',
                  name: 'x',
                  font: { fontWeight: '600', color: 'white' },
                }}
              ></CircularChart3DSeriesDirective>
            </CircularChart3DSeriesCollectionDirective>
          </CircularChart3DComponent>
        </div>
        <div className="col-lg-3 property-section">
          <ButtonComponent
            id="chart-print"
            iconCss="e-icons e-print-icon"
            cssClass="e-flat"
            isPrimary={true}
          >
            Export
          </ButtonComponent>
        </div>
      </div>
    </div>
  );
};
export default Print;

```

```
createRoot(document.getElementById('charts')).render(<Print />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import { createRoot } from 'react-dom/client';
/**
 * Sample for Chart print
 */
import * as React from 'react';
import { useEffect, useRef } from 'react';
import {
  CircularChart3DSeriesDirective,
  CircularChart3DSeriesCollectionDirective,
  Inject,
  CircularChart3DComponent,
  CircularChartDataLabel3D,
  CircularChartLegend3D,
  PieSeries3D,
  CircularChartExport3D,
} from '@syncfusion/ej2-react-charts';
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
export let circularData: any[] = [
  { x: 'John', y: 10, dataLabelMappingName: '$10k' },
  { x: 'Jake', y: 12, dataLabelMappingName: '$12k' },
  { x: 'Peter', y: 18, dataLabelMappingName: '$18k' },
  { x: 'James', y: 11, dataLabelMappingName: '$11k' },
  { x: 'Mary', y: 9.7, dataLabelMappingName: '$9.7k' },
];
const Print = () => {
  useEffect(() => {
    const button = document.getElementById('chart-print');
    button.addEventListener('click', onClick);
  }, []);
  let chartInstance = useRef(null);
  const onClick = (e) => {
    chartInstance.current.export('JPEG', 'chart');
  };
  return (
    <div className="control-pane">
      <div className="control-section row">
        <div className="col-lg-9">
          <CircularChart3DComponent
            id="charts"
            ref={chartInstance}
            title="Browser Market Shares in November 2023"
            tilt={-45}
            enableExport={true}
            legendSettings={{ visible: false, position: 'Right' }}
          >
            <Inject
              services={[
                PieSeries3D,
                CircularChartDataLabel3D,
                CircularChartLegend3D,
```

```

        CircularChartExport3D,
      ]}
    />
    <CircularChart3DSeriesCollectionDirective>
      <CircularChart3DSeriesDirective
        dataSource={circularData}
        xName="x"
        yName="y"
        dataLabel={{
          visible: true,
          position: 'Inside',
          name: 'x',
          font: { fontWeight: '600', color: 'white' },
        }}
      ></CircularChart3DSeriesDirective>
    </CircularChart3DSeriesCollectionDirective>
  </CircularChart3DComponent>
</div>
<div className="col-lg-3 property-section">
  <ButtonComponent
    id="chart-print"
    iconCss="e-icons e-print-icon"
    cssClass="e-flat"
    isPrimary={true}
  >
    Export
  </ButtonComponent>
</div>
</div>
</div>
);
};
export default Print;
createRoot(document.getElementById('charts')).render(<Print />);
{% endraw %}

```

Circular Gauge

Getting Started

This section explains you the steps required to create a simple circular gauge and demonstrate the basic usage of circular gauge control.

Dependencies

Following is the list of minimum dependencies required to use the Circular Gauge.

```
`ts
```

```
|-- @syncfusion/ej2-react-circulargauge
```

```
|-- @syncfusion/ej2-react-base
```

```
|-- @syncfusion/ej2-circulargauge
```

```
|-- @syncfusion/ej2-base
```

```
|-- @syncfusion/ej2-svg-base
```



```
|-- @syncfusion/ej2-paf-export  
,
```

Installation and configuration

To get started with the React application, [create-react-app](#) can be used to setup the application. To install **create-react-app** run the following command.

```
,  
  
npm install -g create-react-app  
,
```

To create basic React application, run the following command.

```
,  
  
create-react-app quickstart  
,
```

Now, the application is created in the **quickstart** folder. Run the following command to navigate to the **quickstart** folder, and install the required **npm** packages.

```
,  
  
cd quickstart  
,
```

In the **quickstart** application, the Syncfusion component is added in the JavaScript file.

Creating a React application with TypeScript

To create React application with TypeScript, use the following command.

```
,  
  
create-react-app quickstart --template typescript  
,
```

Now, the application is created in the **quickstart** folder. Run the following command to navigate to the **quickstart** folder, and install the required **npm** packages.

```
,  
  
cd quickstart  
,
```

Adding Syncfusion packages

All the available Essential JS 2 packages are published in [npmjs.com](#) public registry. To install Circular Gauge package, use the following command.

```
,  
  
npm install @syncfusion/ej2-react-circulargauge --save  
,
```

Adding Circular Gauge component to the Project

Now, the Circular Gauge component can be added in the application. To initialize the Circular Gauge control in the React application, import the Circular Gauge control in the **src/App.js** or **src/App.tsx** as per the application. Please use the below code to include the Circular Gauge component in the application.

```
`ts
import React from 'react';
import { CircularGaugeComponent } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return (<CircularGaugeComponent></CircularGaugeComponent>);
}
export default App;
`
```

Run the application

The Circular Gauge control is now included in the **quickstart** application. Use the following command to run the application.

```
`
npm start
`
```

The below example shows the basic Circular Gauge.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent } from '@syncfusion/ej2-react-circulargauge';
export function App(){
  return(<CircularGaugeComponent ></CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent } from '@syncfusion/ej2-react-circulargauge';
export function App(){
  return(<CircularGaugeComponent ></CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

Set Pointer Value

You can change the pointer value in the above sample using [value](#) property in [pointers](#).

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value={35}></PointerDirective>
          </PointersDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value={35}></PointerDirective>
          </PointersDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

Creating a Next.js Application Using Syncfusion React Components

This section provides a step-by-step guide for setting up a Next.js application and integrating the Syncfusion React Circular Gauge component.

What is Next.js?

[Next.js](#) is a React framework that makes it easy to build fast, SEO-friendly, and user-friendly web applications. It provides features such as server-side rendering, automatic code splitting, routing, and API routes, making it an excellent choice for building modern web applications.

Prerequisites

Before getting started with the Next.js application, ensure the following prerequisites are met:

- [Node.js 18.17](#) or later.
- The application is compatible with macOS, Windows, and Linux operating systems.

Create a Next.js application

To create a new Next.js application, use one of the commands that are specific to either NPM or Yarn.

NPM

```
npx create-next-app@latest
```

YARN

```
yarn create next-app
```

Using one of the above commands will lead you to set up additional configurations for the project as below:

1. Define the project name: Users can specify the name of the project directly. Let's specify the name of the project as `ej2-nextjs-circular-gauge`.

CMD

```
√ What is your project named? » ej2-nextjs-circular-gauge
```

2. Select the required packages.

CMD

```
√ What is your project named? ... ej2-nextjs-circular-gauge
√ Would you like to use TypeScript? ... No / `Yes`
√ Would you like to use ESLint? ... No / `Yes`
√ Would you like to use Tailwind CSS? ... `No` / Yes
√ Would you like to use `src/` directory? ... No / `Yes`
√ Would you like to use App Router? (recommended) ... No / `Yes`
√ Would you like to customize the default import alias? ... `No` / Yes
Creating a new Next.js app in D:\ej2-nextjs-circular-gauge.
```

3. Once complete the above mentioned steps to create `ej2-nextjs-circular-gauge`, navigate to the directory using the below command:

CMD

```
cd ej2-nextjs-circular-gauge
```

The application is ready to run with default settings. Now, let's add Syncfusion components to the project.

Install Syncfusion React packages

Syncfusion React component packages are available at [npmjs.com](https://www.npmjs.com). To use Syncfusion React components in the project, install the corresponding npm package.

Here, the [React Circular Gauge component](#) is used as an example. To install the React Circular Gauge component in the project, use the following command:

NPM

```
npm install @syncfusion/ej2-react-circulargauge --save
```

YARN

```
yarn add @syncfusion/ej2-react-circulargauge
```

Add Syncfusion React component

Follow the below steps to add the React Circular gauge components to the Next.js project:

1. Define the Circular gauge component in the **src/app/page.tsx** file, as shown below:

PAGE.TSX

```
'use client'
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
    <AxesDirective>
    <AxisDirective>
    <PointersDirective>
    <PointerDirective value={35}></PointerDirective>
    </PointersDirective>
    </AxisDirective>
    </AxesDirective>
    </CircularGaugeComponent>);
}
export default App;
```

Run the application

To run the application, use the following command:

NPM

```
npm run dev
```

YARN

```
yarn run dev
```

To learn more about the functionality of the Circular Gauge component, refer to the [documentation](#).

[View the NEXT.js Circular Gauge sample in the GitHub repository.](#)

Gauge dimensions in React Circular gauge component

Size for Container

Circular gauge can render to its container size. You can set the size via inline or CSS as demonstrated below.

```
,  
  
<div id='container'>  
  
<div id='circular-container' style="width:650px; height:350px;"></div>  
  
</div>  
  
,
```

INDEX.JSX

```
{% raw %}  
import * as React from "react";  
import * as ReactDOM from "react-dom";  
import { CircularGaugeComponent } from '@syncfusion/ej2-react-circulargauge';  
export function App() {  
  return(  
    <CircularGaugeComponent >  
      </CircularGaugeComponent>;  
  )  
}  
const root = ReactDOM.createRoot(document.getElementById('container'));  
root.render(<App />);  
{% endraw %}
```

INDEX.TSX

```
{% raw %}  
import * as React from "react";  
import * as ReactDOM from "react-dom";  
import { CircularGaugeComponent } from '@syncfusion/ej2-react-circulargauge';  
export function App() {  
  return(  
    <CircularGaugeComponent >  
      </CircularGaugeComponent>;  
  )  
}  
const root = ReactDOM.createRoot(document.getElementById('container'));  
root.render(<App />);  
{% endraw %}
```

<!-- markdownlint-disable MD036 -->

[Size for Circular Gauge](#)

<!-- markdownlint-disable MD036 -->

You can also set size for the gauge directly through [width](#) and [height](#) properties.

In Pixel

You can set the size of the gauge in pixel as demonstrated below.

<<

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent width='650' height='350'>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent width='650' height='350'>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

In Percentage

By setting value in percentage, gauge gets its dimension with respect to its container. For example, when the height is '50%', gauge renders to half of the container height.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent width='80%' height='50%'>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent width='80%' height='50%'>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

Note: When you do not specify the size, it takes [Link to the Video](#) as the height and window size as its width.

Gauge axes in React Circular Gauge component

By default, gauge will be displayed with an axis. Each axis contains its own ranges, pointers and annotation.

The video below demonstrates how to add and customize axis in the Circular Gauge component.

<!-- markdownlint-disable MD036 -->

Axis Customization

You can customize the width and color of an axis line by using [lineStyle](#) property.

Background for an axis can be customized by using [background](#) property.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective lineStyle = {{
          width: 2,
          color: 'red'
        }} background = 'rgba(0, 128, 128, 0.3)'>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```


INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective lineStyle = {{
          width: 2,
          color: 'red'
        }} background = 'rgba(0, 128, 128, 0.3)' >
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

<!-- markdownlint-disable MD036 -->

Angles and Direction

Circular gauge axis can sweep from 0 to 360 degrees. By default start angle of an axis is 200 degree and end angle is 160 degree and you can customize this option by using [startAngle](#) and [endAngle](#) property.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective startAngle = {270} endAngle = {90}>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
```

```
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective startAngle = {270} endAngle = {90}>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

<!-- markdownlint-disable MD036 -->

The [direction](#) property enables you to render the gauge axis either in **ClockWise** or in **AntiClockWise** direction.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
"@syncfusion/ej2-react-circulargauge";
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective direction = 'AntiClockWise'>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
"@syncfusion/ej2-react-circulargauge";
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective direction = 'AntiClockWise'>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
```

```
{% endraw %}
```

<!-- markdownlint-disable MD036 -->

Axis Radius

By default, radius of an axis is calculated based on the available size. You can customize this, by using [radius](#) property. It takes value either in **percentage** or in **pixel**.

In Pixel

You can set the radius of the gauge in pixel as demonstrated below,

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
"@syncfusion/ej2-react-circulargauge";
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective radius = '150'>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
"@syncfusion/ej2-react-circulargauge";
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective radius = '150'>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

<!-- markdownlint-disable MD036 -->

In Percentage

By setting value in percentage, gauge gets its dimension with respect to its available size. For example, when the radius is '50%', gauge renders to half of the available size.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
 '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective radius = '50%'>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
 '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective radius = '50%'>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

<!-- markdownlint-disable MD036 -->

Ticks

You can customize the [height](#), [color](#) and [width](#) of major ticks and minor ticks by using [majorTicks](#) and [minorTicks](#) property.

By default, [interval](#) for [majorTicks](#) will be calculated automatically and also you can customize the interval for major and minor ticks using [interval](#) property.

INDEX.JSX

```
{% raw %}
```

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
 '@syncfusion/ej2-react-circulargauge';
export function App () {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective majorTicks = {{
          interval: 10,
          color:'red',
          height: 10,
          width: 3
        }} minorTicks = {{
          interval: 5,
          color:'green',
          height: 5,
          width: 2
        }}>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
 '@syncfusion/ej2-react-circulargauge';
export function App () {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective majorTicks = {{
          interval: 10,
          color:'red',
          height: 10,
          width: 3
        }} minorTicks = {{
          interval: 5,
          color:'green',
          height: 5,
          width: 2
        }}>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Tick Position

Both minor and major ticks can be moved by using [offset](#) and

[position](#) property. The [offset](#) defines the distance between the axis and ticks. By default, offset value is 0.

The [position](#) will place the ticks either inside or outside of the axis.

By default, ticks will be placed **inside** the axis.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective majorTicks = {{
          interval: 10,
          color: 'red',
          height: 10,
          width: 3,
          position: 'Inside',
          offset: 5
        }} minorTicks = {{
          interval: 5,
          color: 'green',
          height: 5,
          width: 2,
          position: 'Inside',
          offset: 5
        }} >
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
```

```

    <AxisDirective majorTicks = {{
      interval: 10,
      color: 'red',
      height: 10,
      width: 3,
      position: 'Inside',
      offset: 5
    }} minorTicks = {{
      interval: 5,
      color: 'green',
      height: 5,
      width: 2,
      position: 'Inside',
      offset: 5
    }}>
  </AxisDirective>
</AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% enddraw %}

```

Labels

Labels of an axis can be customized by using [font](#) property in

[labelStyle](#) options.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective labelStyle = {{
          font: {
            color: 'red',
            size: '20px',
            fontWeight: 'Bold'
          }
        }}>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
 '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective labelStyle = {{
          font: {
            color: 'red',
            size: '20px',
            fontWeight: 'Bold'
          }
        }}>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

Label Position

Labels can be moved by using [offset](#) or [position](#) property.

The [offset](#) defines the distance between the labels and ticks.

By default, offset value is 0.

The [position](#) will place the labels either inside or outside of the axis. By default, labels will be placed inside the axis.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
 '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective labelStyle = {{
          position: 'Outside',
          offset: 5
        }}>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```


INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
"@syncfusion/ej2-react-circulargauge";
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective labelStyle = {{
          position: 'Outside',
          offset: 5
        }}>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

Auto Angle

Labels can be swept along the axis angle by enabling [autoAngle](#) property.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
"@syncfusion/ej2-react-circulargauge";
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective labelStyle = {{
          autoAngle: true
        }}>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
```

```
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
 '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective labelStyle = {{
          autoAngle: true
        }}>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

Smart Labels

When an axis makes a complete circle, then the first and last label of the axis will get overlap with each other.

In this scenario, you can either hide 1st or last label using [hiddenLabel](#) property.

When `hiddenLabel` value is `First`, then the 1st label will be hidden and when the `hiddenLabel` value is `'Last'`,

then the last label will be hidden.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
 '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective minimum = {0} maximum = {12} startAngle = {0} endAngle =
{360}
          majorTicks = {{
            interval: 1,
            position: 'Inside',
            height: 10
          }} minorTicks = {{
            interval: 0.2,
            position: 'Inside',
            height: 5
          }} labelStyle = {{
            position: 'Inside',
            hiddenLabel: 'First'
          }}>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
```

```

}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective minimum = {0} maximum = {12} startAngle = {0} endAngle =
{360}
          majorTicks = [{
            interval: 1,
            position: 'Inside',
            height: 10
          }]
          minorTicks = [{
            interval: 0.2,
            position: 'Inside',
            height: 5
          }]
          labelStyle = [{
            position: 'Inside',
            hiddenLabel: 'First'
          }]
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Label Format

Axis labels can be formatted by using [format](#) property in [labelStyle](#) and its supports all globalize format.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective labelStyle = [{
          format: 'p1'
        }]
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}

```

```

    </AxisDirective>
  </AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective labelStyle = {{
          format: 'p1'
        }}>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

The following table describes the result of applying some commonly used label formats on numeric values.

<!-- markdownlint-disable MD033 -->

Label Value	Label Format property value	Result	Description
1000	n1	1000.0	The Number is rounded to 1 decimal place
1000	n2	1000.00	The Number is rounded to 2 decimal place
1000	n3	1000.000	The Number is rounded to 3 decimal place
0.01	p1	1.0%	The Number is converted to percentage with 1 decimal place
0.01	p2	1.00%	The Number is converted to percentage with 2 decimal place
0.01	p3	1.000%	The Number is converted to percentage with 3 decimal place
1000	c1	\$1,000.0	The Currency symbol is appended to number and number is rounded to 1 decimal place

1000	c2	\$1,000.00	The Currency symbol is appended to number and number is rounded to 2 decimal place
------	----	------------	--

Custom Label Format

Axis labels support custom label format using placeholder like {value}°C, in which the value represent the axis label e.g 20°C.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective labelStyle = {{
          format: '{value}°C'
        }}>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective labelStyle = {{
          format: '{value}°C'
        }}>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

Hide intersecting axis labels

When the axis labels overlap with each other, you can hide the intersected labels by setting the `hideIntersectingLabel` property to true in the axis.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective minimum = {0} maximum = {200}
          startAngle = {270} endAngle = {90}
          hideIntersectingLabel= {true}
          majorTicks = {{
            interval: 4
          }}
          minorTicks = {{
            interval: 2
          }}>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective minimum = {0} maximum = {200}
          startAngle = {270} endAngle = {90}
          hideIntersectingLabel= {true}
          majorTicks = {{
            interval: 4
          }}
          minorTicks = {{
            interval: 2
          }}>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
```

```
root.render(<App />);
{% endraw %}
```

Minimum and Maximum

The [minimum](#) and [maximum](#) properties enables you to customize the start and end values of an axis.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective minimum = {50} maximum = {250}>
      </AxisDirective>
    </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective minimum = {50} maximum = {250}>
      </AxisDirective>
    </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

Multiple Axes

In addition to the default axis, you can add n number of axis to a gauge. Each axis will have its own ranges, pointers, annotations and customization options.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
```

```

import { CircularGaugeComponent, AxesDirective, AxisDirective } from
 '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective majorTicks = {{
          interval: 10,
          position: 'Inside',
          height: 10
        }} minorTicks = {{
          interval: 5,
          position: 'Inside',
          height: 5
        }} pointers = {[[]]}>
      </AxisDirective>
      <AxisDirective majorTicks = {{
        interval: 10,
        position: 'Inside',
        height: 10
      }} minorTicks = {{
        interval: 5,
        position: 'Inside',
        height: 5
      }} pointers = {[[]]}>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective, PointerModel }
 from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective majorTicks = {{
          interval: 10,
          position: 'Inside',
          height: 10
        }} minorTicks = {{
          interval: 5,
          position: 'Inside',
          height: 5
        }} pointers = {[[]] as PointerModel[]}>
      </AxisDirective>
      <AxisDirective majorTicks = {{
        interval: 10,

```



```

        position: 'Inside',
        height: 10
    }} minorTicks = {{
        interval: 5,
        position: 'Inside',
        height: 5
    }} pointers = {[[] as PointerModel[]]}>
    </AxisDirective>
</AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Gauge ranges in React Circular gauge component

You can categories certain interval on gauge axis using [ranges](#) property.

Start and End

Start and end value of a range in an axis can be customized by using [start](#) and [end](#) properties.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
RangesDirective, RangeDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
    return(
        <CircularGaugeComponent >
            <AxesDirective>
                <AxisDirective>
                    <RangesDirective>
                        <RangeDirective start = {40} end = {80}></RangeDirective>
                    </RangesDirective>
                </AxisDirective>
            </AxesDirective>
        </CircularGaugeComponent>);
    }
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
RangesDirective, RangeDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
    return(
        <CircularGaugeComponent >
            <AxesDirective>

```

```

    <AxisDirective>
      <RangesDirective>
        <RangeDirective start = {40} end = {80}></RangeDirective>
      </RangesDirective>
    </AxisDirective>
  </AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Customization

Color and thickness of the range can be customized by using [color](#), [startWidth](#) and [endWidth](#) property.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
RangesDirective, RangeDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <RangesDirective>
            <RangeDirective start = {40} end = {80} startWidth = {15}
endWidth = {15}
            color = '#ff5985'></RangeDirective>
          </RangesDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
RangesDirective, RangeDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <RangesDirective>

```

```

        <RangeDirective start = {40} end = {80} startWidth = {15}
endWidth = {15}
        color = '#ff5985'></RangeDirective>
    </RangesDirective>
</AxisDirective>
</AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

<!-- markdownlint-disable MD036 -->

Radius

You can place the range inside or outside of the axis by using [radius](#) property. The radius of the range can take value either in percentage or in pixels. By default, ranges take 100% of the axis radius.

In Pixel

You can set the radius of the range in pixel as demonstrated below,

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
RangesDirective, RangeDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
    return(
        <CircularGaugeComponent >
            <AxesDirective>
                <AxisDirective>
                    <RangesDirective>
                        <RangeDirective start = {40} end = {80} radius =
'100'></RangeDirective>
                    </RangesDirective>
                </AxisDirective>
            </AxesDirective>
        </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
RangesDirective, RangeDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
    return(
        <CircularGaugeComponent >

```

```

    <AxesDirective>
      <AxisDirective>
        <RangesDirective>
          <RangeDirective start = {40} end = {80} radius =
'100'></RangeDirective>
        </RangesDirective>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>;
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

<!-- markdownlint-disable MD036 -->

In Percentage

By setting value in percentage, range gets its dimension with respect to its axis radius. For example, when the radius is '50%', range renders to half of the axis radius.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
RangesDirective, RangeDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <RangesDirective>
            <RangeDirective start = {40} end = {80} radius =
'50%'></RangeDirective>
          </RangesDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
RangesDirective, RangeDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>

```

```

    <AxisDirective>
      <RangesDirective>
        <RangeDirective start = {40} end = {80} radius =
'50%'></RangeDirective>
      </RangesDirective>
    </AxisDirective>
  </AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Dragging range

The ranges can be dragged on the axis values by clicking and dragging the same. To enable or disable the range drag, use the [enableRangeDrag](#) property.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective,
RangesDirective, RangeDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent enableRangeDrag={true} height='250px'
width='250px'>
      <AxesDirective>
        <AxisDirective>
          <RangesDirective>
            <RangeDirective start = {0} end = {100} startWidth={8}
endWidth={8} radius='108%' color='#30B32D'></RangeDirective>
          </RangesDirective>
          <PointersDirective>
            <PointerDirective value = {50}></PointerDirective>
          </PointersDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective,
RangesDirective, RangeDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {

```

```

return (
  <CircularGaugeComponent enableRangeDrag={true} height='250px'
width='250px'>
    <AxesDirective>
      <AxisDirective>
        <RangesDirective>
          <RangeDirective start = {0} end = {100} startWidth={8}
endWidth={8} radius='108%' color='#30B32D'></RangeDirective>
        </RangesDirective>
        <PointersDirective>
          <PointerDirective value = {50}></PointerDirective>
        </PointersDirective>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Multiple Ranges

You can add multiple ranges to an axis with the above customization as demonstrated below.

Note: You can set the range color to axis ticks and labels by enabling `useRangeColor` property in `majorTicks`, `minorTicks` and `labelStyle` object.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
RangesDirective, RangeDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective majorTicks = {{ useRangeColor: true }} minorTicks = {{
useRangeColor: true }} labelStyle = {{ useRangeColor: true }}>
          <RangesDirective>
            <RangeDirective start = {0} end = {25} radius =
'108%'></RangeDirective>
            <RangeDirective start = {25} end = {50} radius =
'70%'></RangeDirective>
            <RangeDirective start = {50} end = {75} radius =
'70%'></RangeDirective>
            <RangeDirective start = {75} end = {100} radius =
'108%'></RangeDirective>
          </RangesDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
RangesDirective, RangeDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective majorTicks = {{ useRangeColor: true }} minorTicks = {{
useRangeColor: true }} labelStyle = {{ useRangeColor: true }}>
          <RangesDirective>
            <RangeDirective start = {0} end = {25} radius =
'108%'></RangeDirective>
            <RangeDirective start = {25} end = {50} radius =
'70%'></RangeDirective>
            <RangeDirective start = {50} end = {75} radius =
'70%'></RangeDirective>
            <RangeDirective start = {75} end = {100} radius =
'108%'></RangeDirective>
          </RangesDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

Rounded corner radius

You can customize the corner radius using the `roundedCornerRadius` property in `ranges`.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
RangesDirective, RangeDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <RangesDirective>
            <RangeDirective start = {40} end = {80} radius = '50%'
roundedCornerRadius = {5}></RangeDirective>
          </RangesDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

```
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
RangesDirective, RangeDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <RangesDirective>
            <RangeDirective start = {40} end = {80} radius = '50%'
roundedCornerRadius = {5}></RangeDirective>
          </RangesDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

Gradient Color

Gradient support allows to add multiple colors in the ranges and pointers of the circular gauge. The following gradient types are supported in the circular gauge.

- Linear Gradient
- Radial Gradient

Linear Gradient

Using linear gradient, colors will be applied in a linear progression. The start value of the linear gradient can be set using the [startValue](#) property. The end value of the linear gradient will be set using the [endValue](#) property. The color stop values such as color, opacity and offset are set using [colorStop](#) property.

To apply linear gradient to the range, follow the below code sample.

INDEX.JSX

```
{% raw %}
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import {
  CircularGaugeComponent,
  AxesDirective,
  AxisDirective,
  RangesDirective,
  RangeDirective,
  Annotations,

```



```

Inject,
Gradient,
PointerDirective,
PointersDirective,
AnnotationsDirective,
AnnotationDirective,
} from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent
      title="Shot Put Distance"
      titleStyle={{ size: '18px' }}
      centerY="57%"
    >
      <Inject services={[Annotations, Gradient]} />
      <AxesDirective>
        <AxisDirective
          startAngle={200}
          endAngle={130}
          radius="90%"
          minimum={0}
          maximum={14}
          lineStyle={{
            width: 0,
            color: '#1d1d1d',
          }}
          majorTicks={{ width: 0 }}
          minorTicks={{ width: 0 }}
          labelStyle={{ font: { size: '0px' } }}
        >
          <PointersDirective>
            <PointerDirective
              type="Marker"
              value={12}
              markerShape="Image"
              imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/football.png"
              radius="108%"
              markerWidth={28}
              markerHeight={28}
              animation={{ duration: 1500 }}
            />
            <PointerDirective
              type="Marker"
              value={11}
              markerShape="Image"
              imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/basketball.png"
              radius="78%"
              markerWidth={28}
              markerHeight={28}
              animation={{ duration: 1200 }}
            />
            <PointerDirective
              type="Marker"
              value={10}
              markerShape="Image"

```

```

        imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/golfball.png"
        radius="48%"
        markerWidth={28}
        markerHeight={28}
        animation={{ duration: 900 }}
    />
    <PointerDirective
        type="Marker"
        value={12}
        markerShape="Image"
        imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/Athletics.png"
        radius="0%"
        markerWidth={90}
        markerHeight={90}
        animation={{ duration: 0 }}
    />
    <PointerDirective
        type="Marker"
        value={0.1}
        markerShape="Image"
        imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/girl.png"
        radius="108%"
        markerWidth={28}
        markerHeight={28}
        animation={{ duration: 1500 }}
    />
    <PointerDirective
        type="Marker"
        value={0.1}
        markerShape="Image"
        imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/man-one.png"
        radius="78%"
        markerWidth={28}
        markerHeight={28}
        animation={{ duration: 1500 }}
    />
    <PointerDirective
        type="Marker"
        value={0.1}
        markerShape="Image"
        imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/man-two.png"
        radius="48%"
        markerWidth={28}
        markerHeight={28}
        animation={{ duration: 1500 }}
    />
</PointersDirective>
<RangesDirective>
    <RangeDirective
        start={0}
        end={12}
        radius="115%"

```

```

        color="#01aebe"
        startWidth={25}
        endWidth={25}
        linearGradient={{
          startValue: '0%',
          endValue: '100%',
          colorStop: [
            { color: '#9E40DC', offset: '0%', opacity: 0.9 },
            { color: '#E63B86', offset: '70%', opacity: 0.9 }
          ]
        }}
      />
    <RangeDirective
      start={0}
      end={11}
      radius="85%"
      color="#3bceac"
      startWidth={25}
      endWidth={25}
      linearGradient={{
        startValue: '0%',
        endValue: '100%',
        colorStop: [
          { color: '#9E40DC', offset: '0%', opacity: 0.9 },
          { color: '#E63B86', offset: '70%', opacity: 0.9 }
        ]
      }}
    />
    <RangeDirective
      start={0}
      end={10}
      radius="55%"
      color="#ee4266"
      startWidth={25}
      endWidth={25}
      linearGradient={{
        startValue: '0%',
        endValue: '100%',
        colorStop: [
          { color: '#9E40DC', offset: '0%', opacity: 0.9 },
          { color: '#E63B86', offset: '70%', opacity: 0.9 }
        ]
      }}
    />
  </RangesDirective>
  <AnnotationsDirective>
    <AnnotationDirective
      content="12 M"
      radius="108%"
      angle={98}
      zIndex="1"
    />
    <AnnotationDirective
      content="11 M"
      radius="80%"
      angle={81}
      zIndex="1"
    />
  </AnnotationsDirective>

```

```

        />
        <AnnotationDirective
            content="10 M"
            radius="50%"
            angle={69}
            zIndex="1"
        />
        <AnnotationDirective
            content="Doe"
            radius="108%"
            angle={190}
            zIndex="1"
        />
        <AnnotationDirective
            content="Almaida"
            radius="80%"
            angle={185}
            zIndex="1"
        />
        <AnnotationDirective
            content="John"
            radius="50%"
            angle={180}
            zIndex="1"
        />
    </AnnotationsDirective>
</AxisDirective>
</AxesDirective>
</CircularGaugeComponent>
);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import {
    CircularGaugeComponent,
    AxesDirective,
    AxisDirective,
    RangesDirective,
    RangeDirective,
    Annotations,
    Inject,
    Gradient,
    PointerDirective,
    PointersDirective,
    AnnotationsDirective,
    AnnotationDirective,
} from '@syncfusion/ej2-react-circulargauge';
export function App() {
    return (

```

```

<CircularGaugeComponent
  title="Shot Put Distance"
  titleStyle={{ size: '18px' }}
  centerY="57%"
>
  <Inject services={[Annotations, Gradient]} />
  <AxesDirective>
    <AxisDirective
      startAngle={200}
      endAngle={130}
      radius="90%"
      minimum={0}
      maximum={14}
      lineStyle={{
        width: 0,
        color: '#1d1d1d',
      }}
      majorTicks={{ width: 0 }}
      minorTicks={{ width: 0 }}
      labelStyle={{ font: { size: '0px' } }}
    >
      <PointersDirective>
        <PointerDirective
          type="Marker"
          value={12}
          markerShape="Image"
          imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/football.png"
          radius="108%"
          markerWidth={28}
          markerHeight={28}
          animation={{ duration: 1500 }}
        />
        <PointerDirective
          type="Marker"
          value={11}
          markerShape="Image"
          imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/basketball.png"
          radius="78%"
          markerWidth={28}
          markerHeight={28}
          animation={{ duration: 1200 }}
        />
        <PointerDirective
          type="Marker"
          value={10}
          markerShape="Image"
          imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/golfball.png"
          radius="48%"
          markerWidth={28}
          markerHeight={28}
          animation={{ duration: 900 }}
        />
        <PointerDirective
          type="Marker"

```

```

        value={12}
        markerShape="Image"
        imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/Athletics.png"
        radius="0%"
        markerWidth={90}
        markerHeight={90}
        animation={{ duration: 0 }}
      />
      <PointerDirective
        type="Marker"
        value={0.1}
        markerShape="Image"
        imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/girl.png"
        radius="108%"
        markerWidth={28}
        markerHeight={28}
        animation={{ duration: 1500 }}
      />
      <PointerDirective
        type="Marker"
        value={0.1}
        markerShape="Image"
        imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/man-one.png"
        radius="78%"
        markerWidth={28}
        markerHeight={28}
        animation={{ duration: 1500 }}
      />
      <PointerDirective
        type="Marker"
        value={0.1}
        markerShape="Image"
        imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/man-two.png"
        radius="48%"
        markerWidth={28}
        markerHeight={28}
        animation={{ duration: 1500 }}
      />
    </PointersDirective>
    <RangesDirective>
      <RangeDirective
        start={0}
        end={12}
        radius="115%"
        color="#01aebc"
        startWidth={25}
        endWidth={25}
        linearGradient={{
          startValue: '0%',
          endValue: '100%',
          colorStop: [
            { color: '#9E40DC', offset: '0%', opacity: 0.9 },
            { color: '#E63B86', offset: '70%', opacity: 0.9 }
          ]
        }}
      />
    </RangesDirective>
  </CircularGauge>

```

```

    ]
  }}
/>
<RangeDirective
  start={0}
  end={11}
  radius="85%"
  color="#3bceac"
  startWidth={25}
  endWidth={25}
  linearGradient={{
    startValue: '0%',
    endValue: '100%',
    colorStop: [
      { color: '#9E40DC', offset: '0%', opacity: 0.9 },
      { color: '#E63B86', offset: '70%', opacity: 0.9 }
    ]
  }}
/>
<RangeDirective
  start={0}
  end={10}
  radius="55%"
  color="#ee4266"
  startWidth={25}
  endWidth={25}
  linearGradient={{
    startValue: '0%',
    endValue: '100%',
    colorStop: [
      { color: '#9E40DC', offset: '0%', opacity: 0.9 },
      { color: '#E63B86', offset: '70%', opacity: 0.9 }
    ]
  }}
/>
</RangesDirective>
<AnnotationsDirective>
  <AnnotationDirective
    content="12 M"
    radius="108%"
    angle={98}
    zIndex="1"
  />
  <AnnotationDirective
    content="11 M"
    radius="80%"
    angle={81}
    zIndex="1"
  />
  <AnnotationDirective
    content="10 M"
    radius="50%"
    angle={69}
    zIndex="1"
  />
  <AnnotationDirective
    content="Doe"

```

```

        radius="108%"
        angle={190}
        zIndex="1"
      />
      <AnnotationDirective
        content="Almaida"
        radius="80%"
        angle={185}
        zIndex="1"
      />
      <AnnotationDirective
        content="John"
        radius="50%"
        angle={180}
        zIndex="1"
      />
    </AnnotationsDirective>
  </AxisDirective>
</AxesDirective>
</CircularGaugeComponent>
);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Radial Gradient

Using radial gradient, colors will be applied in circular progression. The inner circle position of the radial gradient will be set using the [innerPosition](#) property. The outer circle position of the radial gradient can be set using the [outerPosition](#) property. The color stop values such as color, opacity and offset are set using [colorStop](#) property.

To apply radial gradient to the range, follow the below code sample.

INDEX.JSX

```

{% raw %}
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import {
  CircularGaugeComponent,
  AxesDirective,
  AxisDirective,
  RangesDirective,
  RangeDirective,
  Annotations,
  Inject,
  Gradient,
  PointerDirective,
  PointersDirective,
  AnnotationsDirective,
  AnnotationDirective,
} from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent

```



```

    title="Shot Put Distance"
    titleStyle={{ size: '18px' }}
    centerY="57%"
  >
    <Inject services={[Annotations, Gradient]} />
    <AxesDirective>
      <AxisDirective
        startAngle={200}
        endAngle={130}
        radius="90%"
        minimum={0}
        maximum={14}
        lineStyle={{
          width: 0,
          color: '#1d1d1d'
        }}
        majorTicks={{ width: 0 }}
        minorTicks={{ width: 0 }}
        labelStyle={{ font: { size: '0px' } }}
      >
        <PointersDirective>
          <PointerDirective
            type="Marker"
            value={12}
            markerShape="Image"
            imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/football.png"
            radius="108%"
            markerWidth={28}
            markerHeight={28}
            animation={{ duration: 1500 }}
          />
          <PointerDirective
            type="Marker"
            value={11}
            markerShape="Image"
            imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/basketball.png"
            radius="78%"
            markerWidth={28}
            markerHeight={28}
            animation={{ duration: 1200 }}
          />
          <PointerDirective
            type="Marker"
            value={10}
            markerShape="Image"
            imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/golfball.png"
            radius="48%"
            markerWidth={28}
            markerHeight={28}
            animation={{ duration: 900 }}
          />
          <PointerDirective
            type="Marker"
            value={12}

```

```

        markerShape="Image"
        imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/Athletics.png"
        radius="0%"
        markerWidth={90}
        markerHeight={90}
        animation={{ duration: 0 }}
    />
    <PointerDirective
        type="Marker"
        value={0.1}
        markerShape="Image"
        imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/girl.png"
        radius="108%"
        markerWidth={28}
        markerHeight={28}
        animation={{ duration: 1500 }}
    />
    <PointerDirective
        type="Marker"
        value={0.1}
        markerShape="Image"
        imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/man-one.png"
        radius="78%"
        markerWidth={28}
        markerHeight={28}
        animation={{ duration: 1500 }}
    />
    <PointerDirective
        type="Marker"
        value={0.1}
        markerShape="Image"
        imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/man-two.png"
        radius="48%"
        markerWidth={28}
        markerHeight={28}
        animation={{ duration: 1500 }}
    />
</PointersDirective>
<RangesDirective>
    <RangeDirective
        start={0}
        end={12}
        radius="115%"
        color="#01aebe"
        startWidth={25}
        endWidth={25}
        radialGradient={{
            radius: '50%',
            innerPosition: { x: '50%', y: '50%' },
            outerPosition: { x: '50%', y: '50%' },
            colorStop: [
                { color: '#9E40DC', offset: '90%', opacity: 0.9 },
                { color: '#E63B86', offset: '160%', opacity: 0.9 }
            ]
        }}
    />
</RangesDirective>

```

```

    ]
  }}
/>
<RangeDirective
  start={0}
  end={11}
  radius="85%"
  color="#3bceac"
  startWidth={25}
  endWidth={25}
  radialGradient={{
    radius: '50%',
    innerPosition: { x: '50%', y: '50%' },
    outerPosition: { x: '50%', y: '50%' },
    colorStop: [
      { color: '#9E40DC', offset: '90%', opacity: 0.9 },
      { color: '#E63B86', offset: '160%', opacity: 0.9 }
    ]
  }}
/>
<RangeDirective
  start={0}
  end={10}
  radius="55%"
  color="#ee4266"
  startWidth={25}
  endWidth={25}
  radialGradient={{
    radius: '50%',
    innerPosition: { x: '50%', y: '50%' },
    outerPosition: { x: '50%', y: '50%' },
    colorStop: [
      { color: '#9E40DC', offset: '90%', opacity: 0.9 },
      { color: '#E63B86', offset: '160%', opacity: 0.9 }
    ]
  }}
/>
</RangesDirective>
<AnnotationsDirective>
  <AnnotationDirective
    content="12 M"
    radius="108%"
    angle={98}
    zIndex="1"
  />
  <AnnotationDirective
    content="11 M"
    radius="80%"
    angle={81}
    zIndex="1"
  />
  <AnnotationDirective
    content="10 M"
    radius="50%"
    angle={69}
    zIndex="1"
  />

```

```

        <AnnotationDirective
          content="Doe"
          radius="108%"
          angle={190}
          zIndex="1"
        />
        <AnnotationDirective
          content="Almaida"
          radius="80%"
          angle={185}
          zIndex="1"
        />
        <AnnotationDirective
          content="John"
          radius="50%"
          angle={180}
          zIndex="1"
        />
      </AnnotationsDirective>
    </AxisDirective>
  </AxesDirective>
</CircularGaugeComponent>
);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import {
  CircularGaugeComponent,
  AxesDirective,
  AxisDirective,
  RangesDirective,
  RangeDirective,
  Annotations,
  Inject,
  Gradient,
  PointerDirective,
  PointersDirective,
  AnnotationsDirective,
  AnnotationDirective,
} from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent
      title="Shot Put Distance"
      titleStyle={{ size: '18px' }}
      centerY="57%"
    >
      <Inject services={[Annotations, Gradient]} />
      <AxesDirective>

```

```

<AxisDirective
  startAngle={200}
  endAngle={130}
  radius="90%"
  minimum={0}
  maximum={14}
  lineStyle={{
    width: 0,
    color: '#1d1d1d'
  }}
  majorTicks={{ width: 0 }}
  minorTicks={{ width: 0 }}
  labelStyle={{ font: { size: '0px' } }}
>
  <PointersDirective>
    <PointerDirective
      type="Marker"
      value={12}
      markerShape="Image"
      imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/football.png"
      radius="108%"
      markerWidth={28}
      markerHeight={28}
      animation={{ duration: 1500 }}
    />
    <PointerDirective
      type="Marker"
      value={11}
      markerShape="Image"
      imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/basketball.png"
      radius="78%"
      markerWidth={28}
      markerHeight={28}
      animation={{ duration: 1200 }}
    />
    <PointerDirective
      type="Marker"
      value={10}
      markerShape="Image"
      imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/golfball.png"
      radius="48%"
      markerWidth={28}
      markerHeight={28}
      animation={{ duration: 900 }}
    />
    <PointerDirective
      type="Marker"
      value={12}
      markerShape="Image"
      imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/Athletics.png"
      radius="0%"
      markerWidth={90}
      markerHeight={90}
    />
  </PointersDirective>
</AxisDirective>

```

```

        animation={{ duration: 0 }}
      />
      <PointerDirective
        type="Marker"
        value={0.1}
        markerShape="Image"
        imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/girl.png"
        radius="108%"
        markerWidth={28}
        markerHeight={28}
        animation={{ duration: 1500 }}
      />
      <PointerDirective
        type="Marker"
        value={0.1}
        markerShape="Image"
        imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/man-one.png"
        radius="78%"
        markerWidth={28}
        markerHeight={28}
        animation={{ duration: 1500 }}
      />
      <PointerDirective
        type="Marker"
        value={0.1}
        markerShape="Image"
        imageUrl="https://ej2.syncfusion.com/react/demos/src/circular-
gauge/images/man-two.png"
        radius="48%"
        markerWidth={28}
        markerHeight={28}
        animation={{ duration: 1500 }}
      />
    </PointersDirective>
    <RangesDirective>
      <RangeDirective
        start={0}
        end={12}
        radius="115%"
        color="#01aebe"
        startWidth={25}
        endWidth={25}
        radialGradient={{
          radius: '50%',
          innerPosition: { x: '50%', y: '50%' },
          outerPosition: { x: '50%', y: '50%' },
          colorStop: [
            { color: '#9E40DC', offset: '90%', opacity: 0.9 },
            { color: '#E63B86', offset: '160%', opacity: 0.9 }
          ]
        }}
      />
      <RangeDirective
        start={0}
        end={11}

```

```

radius="85%"
color="#3bceac"
startWidth={25}
endWidth={25}
radialGradient={{
  radius: '50%',
  innerPosition: { x: '50%', y: '50%' },
  outerPosition: { x: '50%', y: '50%' },
  colorStop: [
    { color: '#9E40DC', offset: '90%', opacity: 0.9 },
    { color: '#E63B86', offset: '160%', opacity: 0.9 }
  ]
}}
/>
<RangeDirective
  start={0}
  end={10}
  radius="55%"
  color="#ee4266"
  startWidth={25}
  endWidth={25}
  radialGradient={{
    radius: '50%',
    innerPosition: { x: '50%', y: '50%' },
    outerPosition: { x: '50%', y: '50%' },
    colorStop: [
      { color: '#9E40DC', offset: '90%', opacity: 0.9 },
      { color: '#E63B86', offset: '160%', opacity: 0.9 }
    ]
  }}
/>
</RangesDirective>
<AnnotationsDirective>
  <AnnotationDirective
    content="12 M"
    radius="108%"
    angle={98}
    zIndex="1"
  />
  <AnnotationDirective
    content="11 M"
    radius="80%"
    angle={81}
    zIndex="1"
  />
  <AnnotationDirective
    content="10 M"
    radius="50%"
    angle={69}
    zIndex="1"
  />
  <AnnotationDirective
    content="Doe"
    radius="108%"
    angle={190}
    zIndex="1"
  />

```

```

        <AnnotationDirective
            content="Almaida"
            radius="80%"
            angle={185}
            zIndex="1"
        />
        <AnnotationDirective
            content="John"
            radius="50%"
            angle={180}
            zIndex="1"
        />
    </AnnotationsDirective>
</AxisDirective>
</AxesDirective>
</CircularGaugeComponent>
);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% enddraw %}

```

See also

- [Tooltip for Ranges](#)

Gauge pointers in React Circular gauge component

Pointers are used to indicate values on the axis. Value of the pointer can be modified using the [value](#) property.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
    return(
        <CircularGaugeComponent >
            <AxesDirective>
                <AxisDirective>
                    <PointersDirective>
                        <PointerDirective value = {90}></PointerDirective>
                    </PointersDirective>
                </AxisDirective>
            </AxesDirective>
        </CircularGaugeComponent>);
    }
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```


INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {90}></PointerDirective>
          </PointersDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

Gauge supports 3 types of pointers such as **Needle**, **RangeBar** and **Marker**. You can choose any one of the pointer by using [type](#) property.

Needle Pointers

A needle pointer contains three parts, a needle, a cap / knob and a tail. The length of the needle can be customized by using [radius](#) property.

The length of the tail can be customized by using [length](#) property.

The radius of the cap can be customized by using [radius](#)

in cap object. The needle and tail length takes value either in **percentage** or **pixel**.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {90} radius = '50%' cap = {{
              radius: 10
            }} needleTail = {{
              length: '25%'
            }}>
          </PointerDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

```

        </PointersDirective>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {90} radius = '50%' cap = {{
              radius: 10
            }} needleTail = {{
              length: '25%'
            }}>
          </PointerDirective>
        </PointersDirective>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

<!-- markdownlint-disable MD036 -->

Customization

Needle color and width can be customized by using [color](#) and [pointerWidth](#) property.

Cap and tails can be customized by using [cap](#) and [needleTail](#) object.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >

```

```

    <AxesDirective>
      <AxisDirective>
        <PointersDirective>
          <PointerDirective value = {90} radius = '50%' cap = {{
            radius: 15,
            color: 'white',
            border: {
              color: '#007DD1',
              width: 5
            }
          }} needleTail = {{
            length: '22%',
            color: '#007DD1'
          }} color = '#007DD1' pointerWidth = {25}>
        </PointerDirective>
      </PointersDirective>
    </AxisDirective>
  </AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {90} radius = '50%' cap = {{
              radius: 15,
              color: 'white',
              border: {
                color: '#007DD1',
                width: 5
              }
            }} needleTail = {{
              length: '22%',
              color: '#007DD1'
            }} color = '#007DD1' pointerWidth = {25}>
          </PointerDirective>
        </PointersDirective>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));

```

```
root.render(<App />);
{% endraw %}
```

The appearance of the needle pointer can be customized by using [needleStartWidth](#) and [needleEndWidth](#).

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective,
Annotations, AnnotationsDirective, AnnotationDirective, Inject } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <Inject services={[ Annotations ]}/>
      <AxesDirective>
        <AxisDirective startAngle={270} radius='90%' minimum={0}
          maximum= {100}
          endAngle= {90}
          lineStyle= {{ width:3, color:'#1E7145' }}
          labelStyle={{
            position:'Outside',
            font : {size: '0px', color: '#1E7145'}
          }} majorTicks = {{
            width :1,
            height:0,
            interval:100
          }} minorTicks={{
            height :0,
            width:0
          }}
        >
          <PointersDirective>
            <PointerDirective value = {70} pointerWidth= {2}
              needleStartWidth= {4}
              needleEndWidth= {4} radius = '80%' color='green' cap = {{
                radius: 8,
                color: 'green',
                border: {
                  color: '#007DD1',
                  width: 5
                },
              }} needleTail = {{
                length: '0%'
              }}
            >
          </PointerDirective>
        </PointersDirective>
        <AnnotationsDirective>
          <AnnotationDirective angle={180} radius='20%' zIndex='1'
            content='<div style="color:#757575; font-family:Roboto; font-
            size:14px;padding-top: 26px">Customized Needle</div>' />
          </AnnotationDirective>
        </AnnotationsDirective>
      </AxisDirective>
    </CircularGaugeComponent >
  );
}
```

```

    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective,
Annotations, AnnotationsDirective, AnnotationDirective, Inject } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <Inject services={[ Annotations ]}/>
      <AxesDirective>
        <AxisDirective startAngle={270} radius='90%' minimum={0}
          maximum= {100}
          endAngle= {90}
          lineStyle= {{ width:3, color:'#1E7145' }}
          labelStyle={{
            position:'Outside',
            font : {size: '0px', color: '#1E7145'}
          }} majorTicks = {{
            width :1,
            height:0,
            interval:100
          }} minorTicks={{
            height :0,
            width:0
          }}
        >
        <PointersDirective>
          <PointerDirective value = {70} pointerWidth= {2}
            needleStartWidth= {4}
            needleEndWidth= {4} radius = '80%' color='green' cap = {{
              radius: 8,
              color: 'green',
              border: {
                color: '#007DD1',
                width: 5
              },
            }} needleTail = {{
              length: '0%'
            }}
          >
        </PointerDirective>
      </PointersDirective>
      <AnnotationsDirective>
        <AnnotationDirective angle={180} radius='20%' zIndex='1'
          content='<div style="color:#757575; font-family:Roboto; font-
            size:14px;padding-top: 26px">Customized Needle</div>' />
        </AnnotationDirective>
      </AnnotationsDirective>
    </CircularGaugeComponent >
  );
}

```

```

    </AxisDirective>
  </AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

RangeBar Pointer

RangeBar pointer is like ranges in an axis, that can be placed on gauge to mark the pointer value.

RangeBar starts from the beginning of the gauge and ends at the pointer value.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {50} type = 'RangeBar' radius = '60%'>
          </PointerDirective>
        </PointersDirective>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {50} type = 'RangeBar' radius = '60%'>
          </PointerDirective>
        </PointersDirective>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

```

    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Customization

RangeBar can be customized in terms of color, border and thickness by using [color](#), [border](#) and [pointerWidth](#) property.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {50} type = 'RangeBar' radius =
'60%' color = '#007DD1'
              border = {{
                color: 'grey',
                width: 2
              }} pointerWidth = {15}>
          </PointerDirective>
        </PointersDirective>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>

```

```

        <PointerDirective value = {50} type = 'RangeBar' radius =
'60%' color = '#007DD1'
            border = {{
                color: 'grey',
                width: 2
            }} pointerWidth = {15}>
        </PointerDirective>
    </PointersDirective>
</AxisDirective>
</AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Rounded corner for range bar pointer

The start and end pointers of range bar in the circular gauge are rounded to form arc gauges.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
RangesDirective, RangeDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
    return (
        <CircularGaugeComponent >
            <AxesDirective>
                <AxisDirective lineStyle={{ color: 'transparent' }}>
                    <RangesDirective>
                        <RangeDirective start = {0} end = {50} radius = '108%'
roundedCornerRadius = {6}></RangeDirective>
                        <RangeDirective start = {50} end = {100} radius = '108%'
roundedCornerRadius = {6}></RangeDirective>
                    </RangesDirective>
                </AxisDirective>
            </AxesDirective>
        </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
RangesDirective, RangeDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
    return (
        <CircularGaugeComponent >
            <AxesDirective>

```



```

        <AxisDirective lineStyle={{ color: 'transparent' }}>
          <RangesDirective>
            <RangeDirective start = {0} end = {50} radius = '108%'
roundedCornerRadius = {6}></RangeDirective>
            <RangeDirective start = {50} end = {100} radius = '108%'
roundedCornerRadius = {6}></RangeDirective>
          </RangesDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>;
  }
  const root = ReactDOM.createRoot(document.getElementById('container'));
  root.render(<App />);
  {% endraw %}

```

Marker Pointer

Different type of marker shape can be used to mark the pointer value in axis. You can change the marker shape using [markerShape](#) property in pointer. Gauge marker supports [Circle](#), [Rectangle](#), [Triangle](#), [InvertedTriangle](#) and [Diamond](#) shape.

We can use image instead of rendering marker shape to denote the pointer value. It can be achieved by setting [markerShape](#) to Image and assigning image path to [imageUrl](#) in pointer.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {90} type = 'Marker' markerShape =
'InvertedTriangle'
              radius = '100%' markerHeight = {15} markerWidth = {15}>
            </PointerDirective>
          </PointersDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>;
  )
  const root = ReactDOM.createRoot(document.getElementById('container'));
  root.render(<App />);
  {% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";

```

```
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {90} type = 'Marker' markerShape =
'InvertedTriangle'
              radius = '100%' markerHeight = {15} markerWidth = {15}>
          </PointerDirective>
        </PointersDirective>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

Customization

The marker can be customized in terms of color, border, width and height by using [color](#), [border](#), [markerWidth](#) and [markerHeight](#) property in [pointer](#).

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {90} type = 'Marker' markerShape =
'Triangle' radius = '100%'
              color = 'white' border= {{
                color: '#007DD1',
                width: 2
              }} markerHeight = {15} markerWidth = {15}>
          </PointerDirective>
        </PointersDirective>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {90} type = 'Marker' markerShape =
'Triangle' radius = '100%'
              color = 'white' border= {{
                color: '#007DD1',
                width: 2
              }} markerHeight = {15} markerWidth = {15}>
          </PointerDirective>
        </PointersDirective>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

Dragging pointer

The pointers can be dragged over the axis values by clicking and dragging the same. To enable or disable the pointer drag, use the [enablePointerDrag](#) property.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return (
    <CircularGaugeComponent enablePointerDrag={true} height='250px'
width='250px'>
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {50}></PointerDirective>
          </PointersDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
```

```

}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return(
    <CircularGaugeComponent enablePointerDrag={true} height='250px'
width='250px'>
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {50}></PointerDirective>
          </PointersDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Multiple Pointers

In addition to the default pointer, you can add n number of pointer to an axis by using **pointers** property.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {90} type = 'Marker' markerShape =
'InvertedTriangle'
              radius = '100%' markerHeight = {15} markerWidth = {15}>
            </PointerDirective>
            <PointerDirective value = {90} type = 'RangeBar' radius = '60%'
pointerWidth = {10}>
            </PointerDirective>
          </PointersDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent >
  );
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

```

        <PointerDirective value = {90} radius = '60%' pointerWidth = {25}
cap = {{
    radius: 15,
    border: {
        width: 5
    }
    needleTail = {{
        length: '22%'
    }}
}}>
    </PointerDirective>
</PointersDirective>
</AxisDirective>
</AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
    return(
        <CircularGaugeComponent >
            <AxesDirective>
                <AxisDirective>
                    <PointersDirective>
                        <PointerDirective value = {90} type = 'Marker' markerShape =
'InvertedTriangle'
                            radius = '100%' markerHeight = {15} markerWidth = {15}>
                        </PointerDirective>
                        <PointerDirective value = {90} type = 'RangeBar' radius = '60%'
pointerWidth = {10}>
                        </PointerDirective>
                        <PointerDirective value = {90} radius = '60%' pointerWidth = {25}
cap = {{
                            radius: 15,
                            border: {
                                width: 5
                            }
                            needleTail = {{
                                length: '22%'
                            }}
                        }}>
                    </PointerDirective>
                </PointersDirective>
            </AxisDirective>
        </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));

```

```
root.render(<App />);
{% endraw %}
```

Animation

Pointer will get animate on loading the gauge, this can be handled by using [animation](#) property in pointer.

The [enable](#) property in animation allows you to enable or disable the animation.

The [duration](#) property specify the duration of the animation in milliseconds.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {90} animation= {{
              enable: true,
              duration: 1500
            }}></PointerDirective>
          </PointersDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {90} animation= {{
              enable: true,
              duration: 1500
            }}></PointerDirective>
          </PointersDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

```

    </PointersDirective>
  </AxisDirective>
</AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Gradient Color

Gradient support allows to add multiple colors in the ranges and pointers of the circular gauge. The following gradient types are supported in the circular gauge.

- Linear Gradient
- Radial Gradient

Linear Gradient

Using linear gradient, colors will be applied in a linear progression. The start value of the linear gradient can be set using the [startValue](#) property. The end value of the linear gradient will be set using the [endValue](#) property. The color stop values such as color, opacity and offset are set using [colorStop](#) property.

The linear gradient can be applied to all pointer types like marker, range bar and needle. To do so, follow the below code sample.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, Gradient, Inject } from
"@syncfusion/ej2-react-circulargauge";
export function App() {
  return (
    <CircularGaugeComponent >
      <Inject services={[ Gradient ]}/>
      <AxesDirective>
        <AxisDirective radius='90%' startAngle={270} endAngle={90} minimum={0}
maximum={100} lineStyle={{ width: 3, color: '#E63B86'}} labelStyle={{ font: {
size: '0px'}}} majorTicks={{ height: 0 }} minorTicks={{ height: 0}}>
          <PointersDirective>
            <PointerDirective value = {80} animation= {{ enable: true,
duration: 1000 }} radius='80%' markerHeight={5} markerWidth={5}
pointerWidth={10} linearGradient = {{
              startValue: '0%',
              endValue: '100%',
              colorStop: [
                { color: '#FEF3F9', offset: '0%', opacity: 0.9 },
                { color: '#E63B86', offset: '70%', opacity: 0.9 } ]
            }} cap={{
              radius: 8,
              color: 'white',
              border: {

```

```

        color: '#E63B86',
        width: 1
    }
  }} needleTail={{
    length: '20%',
    linearGradient: {
      startValue: '0%',
      endValue: '100%',
      colorStop: [
        { color: '#FEF3F9', offset: '0%', opacity: 0.9 },
        { color: '#E63B86', offset: '70%', opacity: 0.9 }
      ]
    }
  }}</PointerDirective>
  <PointerDirective value = {40} animation= {{ enable: true,
duration: 1000 }} radius='60%' markerHeight={5} markerWidth={5}
pointerWidth={10} linearGradient = {{
  startValue: '0%',
  endValue: '100%',
  colorStop: [
    { color: '#FEF3F9', offset: '0%', opacity: 0.9 },
    { color: '#E63B86', offset: '70%', opacity: 0.9 }
  ]
}} cap={{
  radius: 8,
  color: 'white',
  border: {
    color: '#E63B86',
    width: 1
  }
}} needleTail={{
  length: '20%',
  linearGradient: {
    startValue: '0%',
    endValue: '100%',
    colorStop: [
      { color: '#FEF3F9', offset: '0%', opacity: 0.9 },
      { color: '#E63B86', offset: '70%', opacity: 0.9 }
    ]
  }
}}}}>
</PointerDirective>
</PointersDirective>
</AxisDirective>
</AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, Gradient, Inject } from
'@syncfusion/ej2-react-circulargauge';
export function App() {

```



```

return(
  <CircularGaugeComponent >
    <Inject services={[ Gradient ]}/>
    <AxesDirective>
      <AxisDirective radius='90%' startAngle={270} endAngle={90} minimum={0}
maximum={100} lineStyle={{ width: 3, color: '#E63B86'}} labelStyle={{ font: {
size: '0px'}}} majorTicks={{ height: 0 }} minorTicks={{ height: 0}}>
        <PointersDirective>
          <PointerDirective value = {80} animation= {{ enable: true,
duration: 1000 }} radius='80%' markerHeight={5} markerWidth={5}
pointerWidth={10} linearGradient = {{
            startValue: '0%',
            endValue: '100%',
            colorStop: [
              { color: '#FEF3F9', offset: '0%', opacity: 0.9 },
              { color: '#E63B86', offset: '70%', opacity: 0.9 }]
          }} cap={{
            radius: 8,
            color: 'white',
            border: {
              color: '#E63B86',
              width: 1
            }
          }} needleTail={{
            length: '20%',
            linearGradient: {
              startValue: '0%',
              endValue: '100%',
              colorStop: [
                { color: '#FEF3F9', offset: '0%', opacity: 0.9 },
                { color: '#E63B86', offset: '70%', opacity: 0.9 }]
            }
          }}></PointerDirective>
          <PointerDirective value = {40} animation= {{ enable: true,
duration: 1000 }} radius='60%' markerHeight={5} markerWidth={5}
pointerWidth={10} linearGradient = {{
            startValue: '0%',
            endValue: '100%',
            colorStop: [
              { color: '#FEF3F9', offset: '0%', opacity: 0.9 },
              { color: '#E63B86', offset: '70%', opacity: 0.9 }]
          }} cap={{
            radius: 8,
            color: 'white',
            border: {
              color: '#E63B86',
              width: 1
            }
          }} needleTail={{
            length: '20%',
            linearGradient: {
              startValue: '0%',
              endValue: '100%',
              colorStop: [
                { color: '#FEF3F9', offset: '0%', opacity: 0.9 },
                { color: '#E63B86', offset: '70%', opacity: 0.9 }]
            }
          }}>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent >
  )

```

```

    </PointerDirective>
  </PointersDirective>
</AxisDirective>
</AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Radial Gradient

Using radial gradient, colors will be applied in circular progression. The inner circle position of the radial gradient will be set using the [innerPosition](#) property. The outer circle position of the radial gradient can be set using the [outerPosition](#) property. The color stop values such as color, opacity and offset are set using [colorStop](#) property.

The radial gradient can be applied to all pointer types like marker, range bar and needle. To do so, follow the below code sample.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, Inject, Gradient } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent >
      <Inject services={[ Gradient ]}/>
      <AxesDirective>
        <AxisDirective radius='90%' startAngle={270} endAngle={90} minimum={0}
maximum={100} lineStyle={{ width: 3, color: '#E63B86'}} labelStyle={{ font: {
size: '0px'}}} majorTicks={{ height: 0 }} minorTicks={{ height: 0}}>
          <PointersDirective>
            <PointerDirective value = {80} animation= {{ enable: true,
duration: 1000 }} radius='80%' markerHeight={5} markerWidth={5}
pointerWidth={10} radialGradient = {{
              radius: '50%',
              innerPosition: { x: '50%', y: '50%' },
              outerPosition: { x: '50%', y: '50%' },
              colorStop: [
                { color: '#FEF3F9', offset: '0%', opacity: 0.9 },
                { color: '#E63B86', offset: '60%', opacity: 0.9 } ]
            }} cap={{
              radius: 8,
              color: 'white',
              border: {
                color: '#E63B86',
                width: 1
              }
            }} needleTail={{
              length: '20%',
              radialGradient: {
                radius: '50%',

```

```

        innerPosition: { x: '50%', y: '50%' },
        outerPosition: { x: '50%', y: '50%' },
        colorStop: [
          { color: '#FEF3F9', offset: '0%', opacity: 0.9 },
          { color: '#E63B86', offset: '60%', opacity: 0.9 } ]
      }
    }}></PointerDirective>
    <PointerDirective value = {40} animation= {{ enable: true,
duration: 1000 }} radius='60%' markerHeight={5} markerWidth={5}
pointerWidth={10} radialGradient = {{
  radius: '50%',
  innerPosition: { x: '50%', y: '50%' },
  outerPosition: { x: '50%', y: '50%' },
  colorStop: [
    { color: '#FEF3F9', offset: '0%', opacity: 0.9 },
    { color: '#E63B86', offset: '60%', opacity: 0.9 } ]
}} cap={{
  radius: 8,
  color: 'white',
  border: {
    color: '#E63B86',
    width: 1
  }
}} needleTail={{
  length: '20%',
  radialGradient: {
    radius: '50%',
    innerPosition: { x: '50%', y: '50%' },
    outerPosition: { x: '50%', y: '50%' },
    colorStop: [
      { color: '#FEF3F9', offset: '0%', opacity: 0.9 },
      { color: '#E63B86', offset: '60%', opacity: 0.9 } ]
    }
  }}></PointerDirective>
  </PointersDirective>
</AxisDirective>
</AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, Inject, Gradient } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent >
      <Inject services={[ Gradient ]}/>
      <AxesDirective>

```

```

<AxisDirective radius='90%' startAngle={270} endAngle={90} minimum={0}
maximum={100} lineStyle={{ width: 3, color: '#E63B86'}} labelStyle={{ font: {
size: '0px'}}} majorTicks={{ height: 0 }} minorTicks={{ height: 0}}>
  <PointersDirective>
    <PointerDirective value = {80} animation= {{ enable: true,
duration: 1000 }} radius='80%' markerHeight={5} markerWidth={5}
pointerWidth={10} radialGradient = {{
  radius: '50%',
  innerPosition: { x: '50%', y: '50%' },
  outerPosition: { x: '50%', y: '50%' },
  colorStop: [
    { color: '#FEF3F9', offset: '0%', opacity: 0.9 },
    { color: '#E63B86', offset: '60%', opacity: 0.9 } ]
}} cap={{
  radius: 8,
  color: 'white',
  border: {
    color: '#E63B86',
    width: 1
  }
}} needleTail={{
  length: '20%',
  radialGradient: {
    radius: '50%',
    innerPosition: { x: '50%', y: '50%' },
    outerPosition: { x: '50%', y: '50%' },
    colorStop: [
      { color: '#FEF3F9', offset: '0%', opacity: 0.9 },
      { color: '#E63B86', offset: '60%', opacity: 0.9 } ]
    }
  }}></PointerDirective>
    <PointerDirective value = {40} animation= {{ enable: true,
duration: 1000 }} radius='60%' markerHeight={5} markerWidth={5}
pointerWidth={10} radialGradient = {{
  radius: '50%',
  innerPosition: { x: '50%', y: '50%' },
  outerPosition: { x: '50%', y: '50%' },
  colorStop: [
    { color: '#FEF3F9', offset: '0%', opacity: 0.9 },
    { color: '#E63B86', offset: '60%', opacity: 0.9 } ]
}} cap={{
  radius: 8,
  color: 'white',
  border: {
    color: '#E63B86',
    width: 1
  }
}} needleTail={{
  length: '20%',
  radialGradient: {
    radius: '50%',
    innerPosition: { x: '50%', y: '50%' },
    outerPosition: { x: '50%', y: '50%' },
    colorStop: [
      { color: '#FEF3F9', offset: '0%', opacity: 0.9 },
      { color: '#E63B86', offset: '60%', opacity: 0.9 } ]
    }
  }}
  }

```

```

    }}></PointerDirective>
  </PointersDirective>
</AxisDirective>
</AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Gauge annotations in React Circular gauge component

Annotations are used to mark a specific area of interest in the gauge with texts, shapes or images.

Content

You can place any custom element on the axis area by assigning the id of the element to [content](#) property of [annotation](#) object.

Note: To use annotation feature, we need to inject Annotations module into the services.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, Inject, Annotations,
AnnotationsDirective, AnnotationDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return (
    <CircularGaugeComponent>
      <Inject services={[ Annotations ]}/>
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {50}></PointerDirective>
          </PointersDirective>
          <AnnotationsDirective>
            <AnnotationDirective content='<div><div><span>Pointer Value :
50</span></div></div>' zIndex='1'/>
          </AnnotationsDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, Inject, Annotations,

```

```

AnnotationsDirective, AnnotationDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent>
      <Inject services={[ Annotations ]}/>
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {50}></PointerDirective>
          </PointersDirective>
          <AnnotationsDirective>
            <AnnotationDirective content='<div><div><span>Pointer Value :
50</span></div></div>' zIndex='1'/>
          </AnnotationsDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Position

Annotation can be placed around the axis by using [radius](#)

and [angle](#) property. For example, if the angle is 90 degree and the radius is 110%, then the annotation, will be placed at the right side of the axis.

Radius of the annotation takes value either in pixel or percentage. By setting value in percentage, annotation gets its position with respect to its axis radius.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, Inject, Annotations,
AnnotationsDirective, AnnotationDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent>
      <Inject services={[ Annotations ]}/>
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {50}></PointerDirective>
          </PointersDirective>
          <AnnotationsDirective>
            <AnnotationDirective content='<div><div><span>Pointer Value :
50</span></div></div>' angle= {90} radius = '150%' zIndex='1'/>
          </AnnotationsDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}

```

```

    </CircularGaugeComponent>);
  }
  const root = ReactDOM.createRoot(document.getElementById('container'));
  root.render(<App />);
  {% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, Inject, Annotations,
AnnotationsDirective, AnnotationDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return (
    <CircularGaugeComponent>
      <Inject services={[ Annotations ]}/>
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {50}></PointerDirective>
          </PointersDirective>
          <AnnotationsDirective>
            <AnnotationDirective content='<div><div><span>Pointer Value :
50</span></div></div>' angle= {90} radius = '150%' zIndex='1'/>
          </AnnotationsDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Sub Gauge

As the annotation allows you to place any custom element, we can initialize a gauge to the element and can be used to place that in another gauge.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, Inject, Annotations,
AnnotationsDirective, AnnotationDirective, RangesDirective, RangeDirective }
from '@syncfusion/ej2-react-circulargauge';
import { CircularGauge } from '@syncfusion/ej2-circulargauge';
export function App() {
  function loadRender(args) {
    new CircularGauge(
      {
        background: 'transparent',

```

```

        axes: [
            {
                minimum: 0,
                maximum: 12,
                startAngle: 0,
                endAngle: 360,
                majorTicks: { interval: 3 },
                lineStyle: { width: 0 },
                ranges: [
                    {
                        start: 0,
                        end: 3,
                        startWidth: 5,
                        endWidth: 5,
                        color: 'rgba(29,29,29,0.7)'
                    },
                    {
                        start: 3,
                        end: 12,
                        startWidth: 5,
                        endWidth: 5,
                        color: 'rgba(168,145,102,0.1)'
                    }
                ],
                labelStyle: { hiddenLabel: 'First', offset: -5 },
                pointers: [
                    {
                        pointerWidth: 2,
                        radius: '40%',
                        color: 'rgb(29,29,29)',
                        border: { width: 1, color: 'rgb(29,29,29)' },
                        cap: {
                            color: 'rgb(29,29,29)',
                            radius: 2,
                            border: { width: 0.2, color: 'red' }
                        },
                        needleTail: { length: '0%' },
                        animation: { enable: false }
                    }
                ]
            }
        ],
        '#subGauge'
    );
}

return(
    <CircularGaugeComponent loaded= {loadRender}>
        <Inject services={ [ Annotations ] }/>
        <AxesDirective>
            <AxisDirective minimum= {0} maximum= {12} startAngle= {0} endAngle=
{360} lineStyle= {{ width: 0 }} labelStyle = {{
                hiddenLabel: 'First'
            }}>
            <RangesDirective>
                <RangeDirective start={0} end={3}
color='rgba(29,29,29,0.7)'></RangeDirective>

```



```

    <RangeDirective start={3} end={12}
color='rgba(168,145,102,0.1)'></RangeDirective>
  </RangesDirective>
  <AnnotationsDirective>
    <AnnotationDirective content='<div id="subGauge"
style="width:90px;height:90px;"></div>' angle= {270} radius = '40%'
zIndex='1' />
    <AnnotationDirective content='<div id="time"><span>6:30
PM</span></div>' angle= {90} radius = '40%' zIndex='1' />
  </AnnotationsDirective>
  <PointersDirective>
    <PointerDirective pointerWidth= {5} radius= '40%' value= {6.5}
color= 'rgb(29,29,29)' border = {{ width: 1, color: 'rgb(29,29,29)' }} cap =
{{
      color: 'rgb(29,29,29)',
      radius: 0,
      border: {
        width: 0.2,
        color: 'red'
      }
    }} needleTail = {{
      length: '0%'
    }} animation = {{
      enable: false
    }}></PointerDirective>
    <PointerDirective radius= '60%' value = {6} pointerWidth= {5}
color= 'rgb(29,29,29)' border = {{
      width: 1,
      color: 'rgb(29,29,29)'
    }} cap = {{
      color: 'rgb(29,29,29)',
      radius: 0,
      border: {
        width: 0.2,
        color: 'red'
      }
    }} needleTail = {{
      length: '0%'
    }} animation = {{
      enable: false
    }}></PointerDirective>
    <PointerDirective radius= '70%' pointerWidth= {4} value= {9.8}
color= 'rgba(168,145,102,1)' cap = {{
      color: 'rgba(168,145,102,1)',
      radius: 4,
      border: {
        width: 0.2,
        color: 'rgba(168,145,102,1)'
      }
    }} needleTail = {{
      color: 'rgba(168,145,102,1)',
      length: '20%'
    }} animation = {{
      enable: false,
      duration: 500
    }}></PointerDirective>
  </PointersDirective>

```

```

    </AxisDirective>
  </AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, Inject, Annotations,
AnnotationsDirective, AnnotationDirective, RangesDirective, RangeDirective,
ILoadedEventArgs } from '@syncfusion/ej2-react-circulargauge';
import { CircularGauge } from '@syncfusion/ej2-circulargauge';
export function App() {
  function loadRender(args: ILoadedEventArgs) {
    new CircularGauge(
      {
        background: 'transparent',
        axes: [
          {
            minimum: 0,
            maximum: 12,
            startAngle: 0,
            endAngle: 360,
            majorTicks: { interval: 3 },
            lineStyle: { width: 0 },
            ranges: [
              {
                start: 0,
                end: 3,
                startWidth: 5,
                endWidth: 5,
                color: 'rgba(29,29,29,0.7)'
              },
              {
                start: 3,
                end: 12,
                startWidth: 5,
                endWidth: 5,
                color: 'rgba(168,145,102,0.1)'
              }
            ],
            labelStyle: { hiddenLabel: 'First', offset: -5 },
            pointers: [
              {
                pointerWidth: 2,
                radius: '40%',
                color: 'rgb(29,29,29)',
                border: { width: 1, color: 'rgb(29,29,29)' },
                cap: {
                  color: 'rgb(29,29,29)',

```

```

        radius: 2,
        border: { width: 0.2, color: 'red' }
    },
    needleTail: { length: '0%' },
    animation: { enable: false }
    }
    ]
    }
    },
    '#subGauge'
);
}
return(
<CircularGaugeComponent loaded= {loadRender}>
  <Inject services={[ Annotations ]}/>
  <AxesDirective>
    <AxisDirective minimum= {0} maximum= {12} startAngle= {0} endAngle=
{360} lineStyle= {{ width: 0 }} labelStyle = {{
  hiddenLabel: 'First'
}}>
    <RangesDirective>
      <RangeDirective start={0} end={3}
color='rgba(29,29,29,0.7)'></RangeDirective>
      <RangeDirective start={3} end={12}
color='rgba(168,145,102,0.1)'></RangeDirective>
    </RangesDirective>
    <AnnotationsDirective>
      <AnnotationDirective content='<div id="subGauge"
style="width:90px;height:90px;"></div>' angle= {270} radius = '40%'
zIndex='1'/'>
      <AnnotationDirective content='<div id="time"><span>6:30
PM</span></div>' angle= {90} radius = '40%' zIndex='1'/'>
    </AnnotationsDirective>
    <PointersDirective>
      <PointerDirective pointerWidth= {5} radius= '40%' value= {6.5}
color= 'rgb(29,29,29)' border = {{ width: 1, color: 'rgb(29,29,29)' }} cap =
{{
        color: 'rgb(29,29,29)',
        radius: 0,
        border: {
          width: 0.2,
          color: 'red'
        }
      }} needleTail = {{
        length: '0%'
      }} animation = {{
        enable: false
      }}></Pointer>
      <PointerDirective radius= '60%' value = {6} pointerWidth= {5}
color= 'rgb(29,29,29)' border = {{
        width: 1,
        color: 'rgb(29,29,29)'
      }} cap = {{
        color: 'rgb(29,29,29)',
        radius: 0,
        border: {

```

```

        width: 0.2,
        color: 'red'
      }
    }} needleTail = {{
      length: '0%'
    }} animation = {{
      enable: false
    }}></PointerDirective>
    <PointerDirective radius= '70%' pointerWidth= {4} value= {9.8}
    color= 'rgba(168,145,102,1)' cap = {{
      color: 'rgba(168,145,102,1)',
      radius: 4,
      border: {
        width: 0.2,
        color: 'rgba(168,145,102,1)'
      }
    }} needleTail = {{
      color: 'rgba(168,145,102,1)',
      length: '20%'
    }} animation = {{
      enable: false,
      duration: 500
    }}></PointerDirective>
  </PointersDirective>
</AxisDirective>
</AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% enddraw %}

```

See also

- [Tooltip for Annotation](#)

Animation in React Circular Gauge component

All of the elements in the Circular Gauge, such as the axis lines, ticks, labels, ranges, pointers, and annotations, can be animated sequentially by using the [animationDuration](#) property. The animation for the Circular Gauge is enabled when the `animationDuration` property is set to an appropriate value in milliseconds, providing a smooth rendering effect for the component. If the `animationDuration` property is set to `0`, which is the default value, the animation effect is disabled. If the animation is enabled, the component will behave in the following order.

1. The axis line will be animated in the rendering direction (clockwise or anticlockwise).
2. Each tick line and label will then be animated.
3. If available, ranges will be animated.
4. If available, pointers will be animated in the same way as [pointer animation](#).
5. If available, annotations will be animated.

The animation of the Circular Gauge is demonstrated in the following example.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
  CircularGaugeComponent,
  AxesDirective,
  AxisDirective,
  PointersDirective,
  PointerDirective,
  AnnotationsDirective,
  AnnotationDirective,
  Annotations,
  Inject,
  RangesDirective,
  RangeDirective,
  Gradient,
} from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent
      animationDuration={2000}
      id="gauge"
      background="transparent"
    >
      <Inject services={[Annotations, Gradient]} />
      <AxesDirective>
        <AxisDirective
          radius="80%"
          startAngle={230}
          endAngle={130}
          majorTicks={{ offset: 5 }}
          lineStyle={{ width: 8, color: '#E0E0E0' }}
          minorTicks={{ offset: 5 }}
          labelStyle={{ font: { fontFamily: 'inherit' }, offset: -1 }}
        >
          <AnnotationsDirective>
            <AnnotationDirective
              angle={165}
              radius="35%"
              content='<div style="font-size:18px;margin-left: -20px;margin-top: -12px; color:#9DD55A">${pointers[0].value}</div>'
              zIndex="1"
            />
          </AnnotationsDirective>
          <PointersDirective>
            <PointerDirective
              value={60}
              radius="60%"
              pointerWidth={7}
              color="#c06c84"
              animation={{ enable: true, duration: 500 }}
              cap={{ radius: 8, color: '#c06c84', border: { width: 0 } }}
              needleTail={{ length: '0%' }}
            />
          </PointersDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>
  );
}

```

```

    <RangesDirective>
      <RangeDirective
        start={0}
        end={30}
        color="#E63B86"
        startWidth={22}
        endWidth={22}
        radius="60%"
        linearGradient={{
          startValue: '0%',
          endValue: '100%',
          colorStop: [
            { color: '#9e40dc', offset: '0%', opacity: 1 },
            { color: '#d93c95', offset: '70%', opacity: 1 },
          ],
        }}
      ></RangeDirective>
      <RangeDirective
        start={30}
        end={60}
        color="#E0E0E0"
        startWidth={22}
        endWidth={22}
        radius="60%"
      ></RangeDirective>
    </RangesDirective>
  </AxisDirective>
</AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% enddraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import {
  CircularGaugeComponent,
  AxesDirective,
  AxisDirective,
  PointersDirective,
  PointerDirective,
  AnnotationsDirective,
  AnnotationDirective,
  Annotations,
  Inject,
  RangesDirective,
  RangeDirective,
  Gradient,
} from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent

```

```

animationDuration={2000}
id="gauge"
background="transparent"
>
<Inject services={[Annotations, Gradient]} />
<AxesDirective>
  <AxisDirective
    radius="80%"
    startAngle={230}
    endAngle={130}
    majorTicks={{ offset: 5 }}
    lineStyle={{ width: 8, color: '#E0E0E0' }}
    minorTicks={{ offset: 5 }}
    labelStyle={{ font: { fontFamily: 'inherit' }, offset: -1 }}
  >
    <AnnotationsDirective>
      <AnnotationDirective
        angle={165}
        radius="35%"
        content='<div style="font-size:18px;margin-left: -20px;margin-top: -12px; color:#9DD55A">${pointers[0].value}</div>'
        zIndex="1"
      />
    </AnnotationsDirective>
    <PointersDirective>
      <PointerDirective
        value={60}
        radius="60%"
        pointerWidth={7}
        color="#c06c84"
        animation={{ enable: true, duration: 500 }}
        cap={{ radius: 8, color: '#c06c84', border: { width: 0 } }}
        needleTail={{ length: '0%' }}
      />
    </PointersDirective>
    <RangesDirective>
      <RangeDirective
        start={0}
        end={30}
        color="#E63B86"
        startWidth={22}
        endWidth={22}
        radius="60%"
        linearGradient={{
          startValue: '0%',
          endValue: '100%',
          colorStop: [
            { color: '#9e40dc', offset: '0%', opacity: 1 },
            { color: '#d93c95', offset: '70%', opacity: 1 },
          ],
        }}
      ></RangeDirective>
      <RangeDirective
        start={30}
        end={60}
        color="#E0E0E0"
        startWidth={22}

```

```
        endWidth={22}  
        radius="60%"  
      ></RangeDirective>  
    </RangesDirective>  
  </AxisDirective>  
</AxesDirective>  
</CircularGaugeComponent>);  
}  
const root = ReactDOM.createRoot(document.getElementById('container'));  
root.render(<App />);  
{% enddraw %}
```

Only the pointer of the Circular Gauge can be animated individually, not the axis lines, ticks, labels, ranges, and annotations. You can refer this [link](#) to enable only pointer animation.

Gauge legend in React Circular gauge component

Legend provides valuable information for interpreting what the circular gauge axis range displays, and they can be represented in various colors, shapes, and other identifiers based on the data. It gives a breakdown of what each symbol represents in the axis range of circular gauge.

You can add the legend for circular gauge ranges by setting the visible property of `legendSettings` to true.

Legend customization

Customization option is also provided for the legend shape, alignment, and position.

Position and alignment

The position of the legend is used to place legend in various positions. You can use the `position` property in `legendSettings`. Based on the position, the legend item will be aligned. The following options are available to customize the legend position:

- Top
- Bottom
- Left
- Right
- Custom
- Auto

The legend alignment is used to align the legend items in specific location. You can use the alignment property in `legendSettings` to align the legend items. The following options are available to customize the legend alignment:

- Near
- Center
- Far

The legends can also be positioned to absolute position using the `location.x` and `location.y` properties available in `legendSettings`.

Legend size

The legend size can be modified using the height and width properties in `legendSettings`.

Legend opacity

To specify the transparency for legend shape, set the opacity property in legendSettings.

Legend shape

To change the legend item shape, specify the desired shape in the shape property of the legend. By default, the shape of the legend is circle.

It also supports the following shapes:

- Rectangle
- Diamond
- Triangle
- InvertedTriangle
- Image
- Circle

You can customize a shape using the shapeWidth and shapeHeight properties.

Legend padding

You can control the spacing between the legend items using the padding option of the legend. The default value of padding is 5.

Legend border

You can customize the legend border using the border option in the legend. The legend border can be customized using the border color and width properties.

<!-- markdownlint-disable MD009 -->

Font of the legend text

The font of the legend item text can be customized using the following properties:

- fontFamily
- fontStyle
- fontWeight
- opacity
- color
- size

The following code example shows how to add legend in the gauge.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, Legend, AxesDirective, AxisDirective,
RangesDirective, RangeDirective, Inject } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return (
    <CircularGaugeComponent
      legendSettings={{
        visible: true,
        shapeWidth: 30,
```

```

        shapeHeight:30,
        padding:15,
        border: {
            color:'green',
            width:3
        }
    }>
    <Inject services=[ Legend ]/>
    <AxesDirective>
        <AxisDirective minimum={0} maximum={100} majorTicks = {{ useRangeColor:
true }} minorTicks = {{ useRangeColor: true }} labelStyle = {{ useRangeColor:
true }}>
            <RangesDirective>
                <RangeDirective start = {0} end = {25} radius =
'108%'></RangeDirective>
                <RangeDirective start = {25} end = {50} radius =
'108%'></RangeDirective>
                <RangeDirective start = {50} end = {75} radius =
'108%'></RangeDirective>
                <RangeDirective start = {75} end = {100} radius =
'108%'></RangeDirective>
            </RangesDirective>
        </AxisDirective>
    </AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, Legend, AxesDirective, AxisDirective,
RangesDirective, RangeDirective, Inject } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
    return(
        <CircularGaugeComponent
            legendSettings={{
                visible: true,
                shapeWidth:30,
                shapeHeight:30,
                padding:15,
                border: {
                    color:'green',
                    width:3
                }
            }}>
            <Inject services=[ Legend ]/>
            <AxesDirective>
                <AxisDirective minimum={0} maximum={100} majorTicks = {{ useRangeColor:
true }} minorTicks = {{ useRangeColor: true }} labelStyle = {{ useRangeColor:
true }}>

```

```

        <RangesDirective>
            <RangeDirective start = {0} end = {25} radius =
'108% '></RangeDirective>
            <RangeDirective start = {25} end = {50} radius =
'108% '></RangeDirective>
            <RangeDirective start = {50} end = {75} radius =
'108% '></RangeDirective>
            <RangeDirective start = {75} end = {100} radius =
'108% '></RangeDirective>
        </RangesDirective>
    </AxisDirective>
</AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Toggle option in legend

The toggle option has been provided for legend. So, if you toggle the legend, the given color will be changed to the corresponding circular gauge range. You can enable the toggle option using `toggleVisibility` in the `legendSettings` property.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, Legend, Inject, AxesDirective,
AxisDirective, RangesDirective, RangeDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
    return(
        <CircularGaugeComponent
            legendSettings={{
                visible: true,
                toggleVisibility: true
            }}>
            <Inject services=[ Legend ]/>
            <AxesDirective>
                <AxisDirective minimum={0} maximum={100} majorTicks = {{ useRangeColor:
true }} minorTicks = {{ useRangeColor: true }} labelStyle = {{ useRangeColor:
true }} >
                    <RangesDirective>
                        <RangeDirective start = {0} end = {25} radius =
'108% '></RangeDirective>
                        <RangeDirective start = {25} end = {50} radius =
'108% '></RangeDirective>
                        <RangeDirective start = {50} end = {75} radius =
'108% '></RangeDirective>
                        <RangeDirective start = {75} end = {100} radius =
'108% '></RangeDirective>
                    </RangesDirective>
                </AxisDirective>
            </AxesDirective>
        </CircularGaugeComponent>);

```

```

}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, Legend, Inject, AxesDirective,
AxisDirective, RangesDirective, RangeDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return(
    <CircularGaugeComponent
      legendSettings={{
        visible: true,
        toggleVisibility: true
      }}>
      <Inject services={[ Legend ]}/>
      <AxesDirective>
        <AxisDirective minimum={0} maximum={100} majorTicks = {{ useRangeColor:
true }} minorTicks = {{ useRangeColor: true }} labelStyle = {{ useRangeColor:
true }} >
          <RangesDirective>
            <RangeDirective start = {0} end = {25} radius =
'108%'></RangeDirective>
            <RangeDirective start = {25} end = {50} radius =
'108%'></RangeDirective>
            <RangeDirective start = {50} end = {75} radius =
'108%'></RangeDirective>
            <RangeDirective start = {75} end = {100} radius =
'108%'></RangeDirective>
          </RangesDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Paging support in legend

By default, paging will be enabled if the legend items exceed the legend bounds. You can view each legend item by navigating between the pages using navigation buttons.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, Legend, Inject, AxesDirective,
AxisDirective, RangesDirective, RangeDirective } from '@syncfusion/ej2-react-
circulargauge';

```

```

export function App() {
  return(
    <CircularGaugeComponent
      legendSettings={{
        visible: true,
        height: "50"
      }}>
      <Inject services={[ Legend ]}/>
      <AxesDirective>
        <AxisDirective minimum={0} maximum={100} majorTicks = {{ useRangeColor:
true }} minorTicks = {{ useRangeColor: true }} labelStyle = {{ useRangeColor:
true }}>
          <RangesDirective>
            <RangeDirective start = {0} end = {25} radius =
'108% '></RangeDirective>
            <RangeDirective start = {25} end = {50} radius =
'108% '></RangeDirective>
            <RangeDirective start = {50} end = {75} radius =
'108% '></RangeDirective>
            <RangeDirective start = {75} end = {100} radius =
'108% '></RangeDirective>
          </RangesDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, Legend, Inject, AxesDirective,
AxisDirective, RangesDirective, RangeDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return(
    <CircularGaugeComponent
      legendSettings={{
        visible: true,
        height: "50"
      }}>
      <Inject services={[ Legend ]}/>
      <AxesDirective>
        <AxisDirective minimum={0} maximum={100} majorTicks = {{ useRangeColor:
true }} minorTicks = {{ useRangeColor: true }} labelStyle = {{ useRangeColor:
true }}>
          <RangesDirective>
            <RangeDirective start = {0} end = {25} radius =
'108% '></RangeDirective>
            <RangeDirective start = {25} end = {50} radius =
'108% '></RangeDirective>

```

```

        <RangeDirective start = {50} end = {75} radius =
'108%'></RangeDirective>
        <RangeDirective start = {75} end = {100} radius =
'108%'></RangeDirective>
    </RangesDirective>
</AxisDirective>
</AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Legend text customization

You can customize the legend text using `legendText` property in `ranges`.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, Legend, Inject, AxesDirective,
AxisDirective, RangesDirective, RangeDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
    return(
        <CircularGaugeComponent
            legendSettings={{
                visible: true,
            }}>
            <Inject services={[ Legend ]}/>
            <AxesDirective>
                <AxisDirective minimum={0} maximum={100} majorTicks = {{ useRangeColor:
true }} minorTicks = {{ useRangeColor: true }} labelStyle = {{ useRangeColor:
true }}>
                    <RangesDirective>
                        <RangeDirective start = {0} end = {25} radius = '108%'
legendText= 'light air'></RangeDirective>
                        <RangeDirective start = {25} end = {50} radius = '108%'
legendText= 'light air'></RangeDirective>
                        <RangeDirective start = {50} end = {75} radius = '108%'
legendText= 'light breeze'></RangeDirective>
                        <RangeDirective start = {75} end = {100} radius = '108%'
legendText= "gentle breeze"></RangeDirective>
                    </RangesDirective>
                </AxisDirective>
            </AxesDirective>
        </CircularGaugeComponent>);
    }
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, Legend, Inject, AxesDirective,
AxisDirective, RangesDirective, RangeDirective } from '@syncfusion/ej2-react-
circulargauge';
export function App() {
  return (
    <CircularGaugeComponent
      legendSettings={{
        visible: true,
      }}>
      <Inject services={[ Legend ]}/>
      <AxesDirective>
        <AxisDirective minimum={0} maximum={100} majorTicks = {{ useRangeColor:
true }} minorTicks = {{ useRangeColor: true }} labelStyle = {{ useRangeColor:
true }}>
          <RangesDirective>
            <RangeDirective start = {0} end = {25} radius = '108%'
legendText= 'light air'></RangeDirective>
            <RangeDirective start = {25} end = {50} radius = '108%'
legendText= 'light air'></RangeDirective>
            <RangeDirective start = {50} end = {75} radius = '108%'
legendText= 'light breeze'></RangeDirective>
            <RangeDirective start = {75} end = {100} radius = '108%'
legendText= "gentle breeze"></RangeDirective>
          </RangesDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

legendRendering event will be triggered before rendering each legend item, using this event you can customize needed legend items using following arguments.

Argument name	Description
fill	Specifies the legend shape color
text	Specifies the current legend text
shape	Customize the shape of the legends
name	Specifies the name of the event
cancel	Set to true, to cancel the event status

Gauge user interaction in React Circular gauge component

Tooltip for pointers

Circular gauge will displays the pointer details through [tooltip](#), when the mouse is moved over the pointer.

<!-- markdownlint-disable MD036 -->

Tooltip for ranges

Circular gauge displays the information about the ranges through tooltip when hovering the mouse over the ranges. You can enable this feature by setting the type property of tooltip to 'Range' in the array collection.

Tooltip customization for ranges

To customize the range tooltip, use the `rangeSettings` property in tooltip. The following options are available to customize the range tooltip:

- `fill` - Specifies the range tooltip fill color.
- `textStyle` - Specifies the range tooltip text style.
- `format` - Specifies the range content format.
- `template` - Specifies the custom template for tooltip.
- `enableAnimation` - Animates as it moves from one point to another.
- `border` - Specifies the tooltip border.
- `showMouseAtPosition` - Displays the position of the tooltip on the cursor position.

Tooltip for annotation

Circular gauge displays the information about the annotations through tooltip when hovering the mouse over the annotation. You can enable this feature by setting the type property of tooltip to 'Annotation' in the array collection.

Tooltip customization for annotation

To customize the annotation tooltip, use the `annotationSettings` property in tooltip. The following options are available to customize the annotation tooltip:

- `fill` - Specifies the annotation tooltip fill color.
- `textStyle` - Specifies the annotation tooltip text style.
- `format` - Specifies the annotation content format.
- `template` - Specifies the tooltip content with custom template.
- `enableAnimation` - Animates as it moves from one point to another.
- `border` - Specifies the tooltip border.

The following code example shows the tooltip for the pointers, ranges and annotation.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, RangesDirective, Annotations,
AnnotationsDirective, AnnotationDirective, RangeDirective, AxesDirective,
AxisDirective, PointersDirective, PointerDirective, Inject, GaugeTooltip }
from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return ( <CircularGaugeComponent tooltip={{
    type: ['Pointer', 'Range', 'Annotation'],
    enable: true,
    enableAnimation: false,
```



```

        annotationSettings: {
template: '<div>CircularGauge</div>' },
        rangeSettings: { fill: 'red' }
    }}
    >
    <Inject services=[ GaugeTooltip, Annotations ]/>
    <AxesDirective>
        <AxisDirective startAngle={240} endAngle={120} radius='90%'
minimum={0} maximum={120}
        majorTicks={{
            color: 'white', offset: -5, height: 12
        }}
        lineStyle={{ width: 0 }}
        minorTicks={{
            width: 0
        }} labelStyle={{
            useRangeColor: true, font: { color: '#424242',
size: '13px', fontFamily: 'Roboto' }
        }}>
        <AnnotationsDirective>
            <AnnotationDirective content='Circular Gauge'
angle= {180} zIndex= '1' radius = '40%' />
        </AnnotationsDirective>
        <PointersDirective>
            <PointerDirective value={70} radius='60%'
                cap={{
                    radius: 10, border: { color: '#33BCBD',
width: 5 }
                }}
            animation={{
                enable: true, duration: 1500
            }} color='#33BCBD' />
        </PointersDirective>
        <RangesDirective>
            <RangeDirective start={0} end={50} radius='102%'
color='#3A5DC8' startWidth={10} endWidth={10} />
            <RangeDirective start={50} end={120} radius='102%'
color='#33BCBD' startWidth={10} endWidth={10} />
        </RangesDirective>
        </AxisDirective>
    </AxesDirective>
</CircularGaugeComponent>;
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, RangesDirective, Annotations,
AnnotationsDirective, AnnotationDirective, RangeDirective, AxesDirective,
AxisDirective, PointersDirective, PointerDirective, Inject, GaugeTooltip }
from '@syncfusion/ej2-react-circulargauge';

```

```

export function App() {
  return ( <CircularGaugeComponent tooltip={{
    type:['Pointer', 'Range', 'Annotation'],
    enable: true,
    enableAnimation: false,
    annotationSettings: {
template:'<div>CircularGauge</div>' },
    rangeSettings: { fill:'red' }
    }}
    >
    <Inject services={[ GaugeTooltip, Annotations ]}/>
    <AxesDirective>
      <AxisDirective startAngle={240} endAngle={120} radius='90%'
minimum={0} maximum={120}
      majorTicks={{
        color: 'white', offset: -5, height: 12
      }}
      lineStyle={{ width: 0 }}
      minorTicks={{
        width: 0
      }} labelStyle={{
        useRangeColor: true, font: { color: '#424242',
size: '13px', fontFamily: 'Roboto' }
      }}>
        <AnnotationsDirective>
          <AnnotationDirective content='Circular Gauge'
angle= {180} zIndex= '1' radius = '40%' />
        </AnnotationsDirective>
        <PointersDirective>
          <PointerDirective value={70} radius='60%'
            cap={{
              radius: 10, border: { color: '#33BCBD',
width: 5 }
            }}
            animation={{
              enable: true, duration: 1500
            }} color='#33BCBD' />
        </PointersDirective>
        <RangesDirective>
          <RangeDirective start={0} end={50} radius='102%'
color='#3A5DC8' startWidth={10} endWidth={10} />
          <RangeDirective start={50} end={120} radius='102%'
color='#33BCBD' startWidth={10} endWidth={10} />
        </RangesDirective>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

<!-- markdownlint-disable MD036 -->

Template

Any HTML elements can be displayed in the tooltip by using the [template](#) property of the tooltip.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, Inject, GaugeTooltip } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent tooltip= {{
      enable: true,
      template: '<div id="templateWrap"><div style="float: right; padding-
left:10px; line-height:30px;"><span>Pointer &#160;&#160;:&#160;
${value}</span></div></div>'
    }}>
    <Inject services=[ GaugeTooltip ]/>
    <AxesDirective>
      <AxisDirective>
        <PointersDirective>
          <PointerDirective value = {70}></PointerDirective>
        </PointersDirective>
      </AxisDirective>
    </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, Inject, GaugeTooltip } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent tooltip= {{
      enable: true,
      template: '<div id="templateWrap"><div style="float: right; padding-
left:10px; line-height:30px;"><span>Pointer &#160;&#160;:&#160;
${value}</span></div></div>'
    }}>
    <Inject services=[ GaugeTooltip ]/>
    <AxesDirective>
      <AxisDirective>
        <PointersDirective>
          <PointerDirective value = {70}></PointerDirective>
        </PointersDirective>
      </AxisDirective>
    </AxesDirective>
    </CircularGaugeComponent>);
}
```

```

}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Pointer Drag

Pointers can be dragged over the axis value. This can be achieved by clicking and dragging the pointer. To enable or disable the pointer drag, you can use [enablePointerDrag](#) property.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, Inject, GaugeTooltip } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent enablePointerDrag = {true} tooltip= {{
      enable: true,
      template: '<div id="templateWrap"><div style="float: right; padding-
left:10px; line-height:30px;"><span>Pointer &#160;&#160;:&#160;
${value}</span></div></div>'
    }}>
      <Inject services={[ GaugeTooltip ]}/>
      <AxesDirective>
        <AxisDirective>
          <PointersDirective>
            <PointerDirective value = {70}></PointerDirective>
          </PointersDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, Inject, GaugeTooltip } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent enablePointerDrag = {true} tooltip= {{
      enable: true,
      template: '<div id="templateWrap"><div style="float: right; padding-
left:10px; line-height:30px;"><span>Pointer &#160;&#160;:&#160;
${value}</span></div></div>'
    }}>

```

```

    <Inject services={[ GaugeTooltip ]}/>
    <AxesDirective>
      <AxisDirective>
        <PointersDirective>
          <PointerDirective value = {70}></PointerDirective>
        </PointersDirective>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>;
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Gauge print and export in React Circular gauge component

Print

To use the print functionality, we should inject the **Print** module into **services** and set the **allowPrint** property to **true**. The rendered circular gauge can be printed directly from the browser by calling the method [print](#).

INDEX.JSX

```

{% raw %}
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
import { CircularGaugeComponent, Print, Inject } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  var gaugeInstance;
  function clickHandler() {
    gaugeInstance.print();
  }
  return (<div>
    <ButtonComponent onClick={clickHandler}>
      print
    </ButtonComponent>
    <CircularGaugeComponent
      id="circulargauge"
      allowPrint={true}
      ref={(g) => (gaugeInstance = g)}
    >
      <Inject services={[Print]} />
    </CircularGaugeComponent>
  </div>
  );
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from 'react';

```

```
import * as ReactDOM from 'react-dom';
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
import { CircularGaugeComponent, Print, Inject } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  let gaugeInstance : CircularGaugeComponent;
  function clickHandler() {
    gaugeInstance.print();
  }
  return (<div>
    <ButtonComponent onClick={clickHandler}>
      print
    </ButtonComponent>
    <CircularGaugeComponent
      id="circulargauge"
      allowPrint={true}
      ref={ (g) => (gaugeInstance = g) }
    >
      <Inject services={[Print]} />
    </CircularGaugeComponent>
  </div>
  );
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

Export

Image Export

To use the image export functionality, we should inject the `ImageExport` module into `services` and set the `allowImageExport` property to `true`. The rendered circular gauge can be exported as an image using the `export` method. The method requires two parameters: image type and file name. The circular gauge can be exported as an image in the following formats.

- JPEG
- PNG
- SVG

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
import { CircularGaugeComponent, ImageExport, Inject } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  var gaugeInstance;
  function clickHandler() {
    gaugeInstance.export('PNG', 'Gauge');
  }
  return (<div>
    <ButtonComponent onClick= {clickHandler}>Export</ButtonComponent>
  </div>
  );
}
```

```

    <CircularGaugeComponent allowImageExport={true} ref={g => gaugeInstance
= g}>
      <Inject services={[ImageExport]} />
    </CircularGaugeComponent>
  </div>;
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
import { CircularGaugeComponent, ImageExport, Inject } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  let gaugeInstance : CircularGaugeComponent;
  function clickHandler() {
    gaugeInstance.export('PNG', 'Gauge');
  }
  return (<div>
    <ButtonComponent onClick= {clickHandler}>Export</ButtonComponent>
    <CircularGaugeComponent allowImageExport={true} ref={g => gaugeInstance
= g}>
      <Inject services={[ImageExport]} />
    </CircularGaugeComponent>
  </div>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

We can get the image file as base64 string for the JPEG and PNG formats. The circular gauge can be exported to image as a base64 string using the [export](#) method. There are four parameters required: image type, file name, orientation of the exported PDF document which must be set as **null** for image export and finally **allowDownload** which should be set as **false** to return base64 string.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
import { PdfPageOrientation } from '@syncfusion/ej2-pdf-export';
import { CircularGaugeComponent, ImageExport, Inject } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  var gaugeInstance;
  function clickHandler() {
    gaugeInstance.export('PNG', 'Gauge', PdfPageOrientation.Landscape,
false).then((data)=>{
      document.writeln(data);
    });
  }
}

```

```

    })
  }
  return (<div>
    <ButtonComponent onClick= { clickHandler}>Export</ButtonComponent>
    <CircularGaugeComponent allowImageExport={true} ref={g => gaugeInstance
= g}>
      <Inject services={[ImageExport]} />
    </CircularGaugeComponent>
  </div>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
import { CircularGaugeComponent, ImageExport, Inject } from '@syncfusion/ej2-
react-circulargauge';
export function App() {
  let gaugeInstance: CircularGaugeComponent;
  function clickHandler() {
    gaugeInstance.export('PNG', 'Gauge', null, false).then((data)=>{
      document.writeln(data);
    })
  }
  return (<div>
    <ButtonComponent onClick= { clickHandler}>Export</ButtonComponent>
    <CircularGaugeComponent allowImageExport={true} ref={g => gaugeInstance
= g}>
      <Inject services={[ImageExport]} />
    </CircularGaugeComponent>
  </div>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

PDF Export

To use the PDF export functionality, we should inject the PdfExport module into services and set the [allowPdfExport](#) property to **true**. The rendered circular gauge can be exported as PDF using the [export](#) method. The [export](#) method requires three parameters: file type, file name and orientation of the PDF document. The orientation setting is optional and "0" indicates portrait and "1" indicates landscape.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
import { CircularGaugeComponent, PdfExport, Inject } from '@syncfusion/ej2-
react-circulargauge';

```



```

export function App() {
  var gaugeInstance;
  function clickHandler() {
    gaugeInstance.export('PDF', 'Gauge', 0);
  }
  return (<div>
    <ButtonComponent onClick= {clickHandler}>Export</ButtonComponent>
    <CircularGaugeComponent allowPdfExport={true} ref={g => gaugeInstance =
g}>
      <Inject services={[PdfExport]} />
    </CircularGaugeComponent>
  </div>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
import { CircularGaugeComponent, PdfExport, Inject } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  let gaugeInstance : CircularGaugeComponent;
  function clickHandler() {
    gaugeInstance.export('PDF', 'Gauge', 0);
  }
  return (<div>
    <ButtonComponent onClick= {clickHandler}>Export</ButtonComponent>
    <CircularGaugeComponent allowPdfExport={true} ref={g => gaugeInstance =
g}>
      <Inject services={[PdfExport]} />
    </CircularGaugeComponent>
  </div>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Note: The exporting of the circular gauge as base64 string is not supported in the PDF export.

Gauge appearance in React Circular gauge component

Gauge Title

Circular gauge can be given a title by using [title](#) property, to show the information about the gauge. Title can be customized by using [titleStyle](#) property in gauge.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent } from '@syncfusion/ej2-react-circulargauge';

```

```
export function App() {
  return(
    <CircularGaugeComponent title= 'Speedometer'
      titleStyle= {{
        color: '#27d5ff'
      }}>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent title= 'Speedometer'
      titleStyle= {{
        color: '#27d5ff'
      }}>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

Gauge Position

<!-- markdownlint-disable MD036 -->

Gauge can be positioned anywhere in the container with the help of [centerX](#) and [centerY](#) property and it accepts values either in percentage or in pixels.

The default value of the [centerX](#) and [centerY](#) property is 50%, which means gauge will get rendered to the centre of the container.

In Pixel

You can set the mid point of the gauge in pixel as demonstrated below,

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent centerX= '20' centerY= '20'>
      <AxesDirective>
        <AxisDirective lineStyle= {{
          width: 2,

```

```

        color: '#F8F8F8'
      }} startAngle= {90} endAngle= {180}>
    </AxisDirective>
  </AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
"@syncfusion/ej2-react-circulargauge";
export function App() {
  return (
    <CircularGaugeComponent centerX= '20' centerY= '20'>
      <AxesDirective>
        <AxisDirective lineStyle= {{
          width: 2,
          color: '#F8F8F8'
        }} startAngle= {90} endAngle= {180}>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

<!-- markdownlint-disable MD036 -->

In Percentage

By setting the value in percentage, gauge gets its mid point with respect to its plot area. For example, when the [centerX](#) value as '0%' and [centerY](#) value is '50%', gauge will get positioned at the top left corner of the plot area.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
"@syncfusion/ej2-react-circulargauge";
export function App() {
  return (
    <CircularGaugeComponent centerX= '10%' centerY= '50%'>
      <AxesDirective>
        <AxisDirective lineStyle= {{
          width: 2,
          color: '#F8F8F8'
        }} startAngle= {0} endAngle= {180}>

```

```

    </AxisDirective>
  </AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent centerX= '10%' centerY= '50%'>
      <AxesDirective>
        <AxisDirective lineStyle= {{
          width: 2,
          color: '#F8F8F8'
        }} startAngle= {0} endAngle= {180}>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

<!-- markdownlint-disable MD036 -->

Area Customization

Customize the gauge background

Using [background](#) and [border](#) properties, you can change the background color and border of the circular gauge.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, RangesDirective, RangeDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent background= 'skyblue' border= {{
      color: "#FF0000", width: 2
    }}>
      <AxesDirective>
        <AxisDirective radius= '90%' maximum= {120} startAngle= {230}
endAngle= {130} majorTicks= {{
          width: 1, color: '#8c8c8c'

```

```

    }} lineStyle= {{ width: 2 }} minorTicks= {{
      width: 1, color: '#8c8c8c'
    }}>
    <PointersDirective>
      <PointerDirective value= {60} radius= '60%'>
      </PointerDirective>
    </PointersDirective>
    <RangesDirective>
      <RangeDirective start= {0} end= {70} radius=
'110%'></RangeDirective>
      <RangeDirective start= {70} end= {110} radius=
'110%'></RangeDirective>
      <RangeDirective start= {110} end= {120} radius=
'110%'></RangeDirective>
    </RangesDirective>
  </AxisDirective>
</AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, RangesDirective, RangeDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent background= 'skyblue' border= {{
      color: "#FF0000", width: 2
    }}>
      <AxesDirective>
        <AxisDirective radius= '90%' maximum= {120} startAngle= {230}
endAngle= {130} majorTicks= {{
          width: 1, color: '#8c8c8c'
        }} lineStyle= {{ width: 2 }} minorTicks= {{
          width: 1, color: '#8c8c8c'
        }}>
          <PointersDirective>
            <PointerDirective value= {60} radius= '60%'>
            </PointerDirective>
          </PointersDirective>
          <RangesDirective>
            <RangeDirective start= {0} end= {70} radius=
'110%'></RangeDirective>
            <RangeDirective start= {70} end= {110} radius=
'110%'></RangeDirective>
            <RangeDirective start= {110} end= {120} radius=
'110%'></RangeDirective>
          </RangesDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>
  );
}

```

```

    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Gauge Margin

You can set margin for gauge from its container through [margin](#) property.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, RangesDirective, RangeDirective } from
"@syncfusion/ej2-react-circulargauge";
export function App() {
  return (
    <CircularGaugeComponent background= 'skyblue' border= {{
      color: "#FF0000", width: 2
    }} margin= {{
      left: 40, right: 40, top: 40, bottom: 40
    }}>
      <AxesDirective>
        <AxisDirective radius= '90%' maximum= {120} startAngle= {230}
endAngle= {130} majorTicks= {{
          width: 1, color: '#8c8c8c'
        }} lineStyle= {{ width: 2 }} minorTicks= {{
          width: 1, color: '#8c8c8c'
        }}>
          <PointersDirective>
            <PointerDirective value= {60} radius= '60%'>
              </PointerDirective>
            </PointersDirective>
            <RangesDirective>
              <RangeDirective start= {0} end= {70} radius=
'110%'></RangeDirective>
              <RangeDirective start= {70} end= {110} radius=
'110%'></RangeDirective>
              <RangeDirective start= {110} end= {120} radius=
'110%'></RangeDirective>
            </RangesDirective>
          </AxisDirective>
        </AxesDirective>
      </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}

```

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, RangesDirective, RangeDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent background= 'skyblue' border= {{
      color: "#FF0000", width: 2
    }} margin= {{
      left: 40, right: 40, top: 40, bottom: 40
    }}>
      <AxesDirective>
        <AxisDirective radius= '90%' maximum= {120} startAngle= {230}
endAngle= {130} majorTicks= {{
          width: 1, color: '#8c8c8c'
        }} lineStyle= {{ width: 2 }} minorTicks= {{
          width: 1, color: '#8c8c8c'
        }}>
          <PointersDirective>
            <PointerDirective value= {60} radius= '60%'>
            </PointerDirective>
          </PointersDirective>
          <RangesDirective>
            <RangeDirective start= {0} end= {70} radius=
'110%'></RangeDirective>
            <RangeDirective start= {70} end= {110} radius=
'110%'></RangeDirective>
            <RangeDirective start= {110} end= {120} radius=
'110%'></RangeDirective>
          </RangesDirective>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>;
  )
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Radius calculation based on angles

Render semi or quarter circular gauges by modifying the start and end angles. By enabling the radius based on angle option, the radius of circular gauge will be calculated based on the start and end angles to avoid excess white space.

INDEX.JSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent>
      <AxesDirective>
        <AxisDirective lineStyle= {{

```

```
        width: 2,
        color: '#F8F8F8'
      }} startAngle= {270} endAngle= {90}>
    </AxisDirective>
  </AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent>
      <AxesDirective>
        <AxisDirective lineStyle= {{
          width: 2,
          color: '#F8F8F8'
        }} startAngle= {270} endAngle= {90}>
        </AxisDirective>
      </AxesDirective>
    </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

Accessibility in React Circular Gauge component

Circular Gauge has built-in accessibility features like screen reading and WAI-ARIA attributes.

WAI-ARIA attributes

The Circular Gauge component followed the [WAI-ARIA](#) patterns to meet the accessibility. The following ARIA attributes are used in the Circular Gauge component:

| Attributes | Purpose |

| --- | --- |

| **role=region** | It is specified in the pointer where the interactive drag and drop function is supported to update the pointer value. |

| **aria-label** | Provides an accessible name for the axis labels, legend title, legend item label, text pointer and annotation. |

Screen reading in Circular Gauge

Accessibility in the Circular Gauge component ensures that all users, regardless of ability or disability, can use screen reading. The following Circular Gauge elements will be read aloud using screen reading software, such as Narrator for Windows.

Elements	Description
---	---
Axis labels	Reads the axis labels of the Circular Gauge.
Legend title	Reads the title of the legend in the Circular Gauge.
Legend item label	Reads the label of the legend item in the Circular Gauge.
Text pointer	Reads the text content shown as a pointer in Circular Gauge.
Annotation	Reads the content specified in the annotation.

Ensuring accessibility

The Circular Gauge component's accessibility levels are ensured through an [accessibility-checker](#) and [axe-core](#) software tools during automated testing.

The accessibility compliance of the Circular Gauge component is shown in the following sample. Open the [sample](#) in a new window to evaluate the accessibility of the Circular Gauge component with accessibility tools.

See also

- [Accessibility in Syncfusion React components](#)

Internationalization in React Circular Gauge component

Circular Gauge provides internationalization support for below elements.

- Axis Labels
- Tooltip

For more information about number formatter, you can refer [internationalization](#).

Globalization

Globalization is the process of designing and developing a component that works in different cultures/locales.

Internationalization library is used to globalize number in Circular Gauge component using [format](#) property in [labelStyle](#).

<!-- markdownlint-disable MD036 -->

Numeric Format

In the below example, axis labels are globalized to **EUR**.

INDEX.JSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
"@syncfusion/ej2-react-circulargauge";
import { setCulture, setCurrencyCode } from "@syncfusion/ej2-base";
setCulture('de');
setCurrencyCode('EUR');
```

```
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective labelStyle = {{
          //Label format set as currency.
          format: 'c'
        }}>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective } from
'@syncfusion/ej2-react-circulargauge';
import { setCulture, setCurrencyCode } from '@syncfusion/ej2-base';
setCulture('de');
setCurrencyCode('EUR');
export function App() {
  return(
    <CircularGaugeComponent >
      <AxesDirective>
        <AxisDirective labelStyle = {{
          //Label format set as currency.
          format: 'c'
        }}>
      </AxisDirective>
    </AxesDirective>
  </CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}
```

Right-to-left

Circular Gauge can render its elements from right to left, which improves the user experience for certain language users. To do so, set the [enableRtl](#) property to **true**. When this property is enabled, elements such as the tooltip and legend will be rendered from right to left. Meanwhile, the axis can be rendered from right to left by setting the [direction](#) property to **AntiClockWise**. For more information on axis, click [here](#).

The following example illustrates the right to left rendering of the Circular Gauge.

INDEX.JSX

```
{% raw %}
import * as React from "react";
```

```

import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
PointersDirective, PointerDirective, RangesDirective, RangeDirective,
GaugeTooltip, Legend, Inject } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return(
    <CircularGaugeComponent
      enableRtl="true"
      tooltip={{
        type: ['Pointer', 'Range'],
        format: 'Pointer : {value} ',
        enable: true,
        enableAnimation: false,
      }}
      legendSettings={{
        visible: true,
      }}
    >
    <Inject services=[[GaugeTooltip, Legend]] />
    <AxesDirective>
      <AxisDirective
        minimum={0}
        maximum={120}
        startAngle={210}
        endAngle={150}
        radius="80%"
        direction="AntiClockWise"
        majorTicks={{
          height: 10,
          offset: 5,
          color: '#9E9E9E',
        }}
        minorTicks={{
          height: 0,
        }}
        lineStyle={{ width: 10, color: 'transparent' }}
        labelStyle={{
          position: 'Inside',
          useRangeColor: false,
          font: {
            size: '12px',
            color: '#424242',
            fontFamily: 'Roboto',
            fontStyle: 'Regular',
          },
        }},
      >
      <RangesDirective>
        <RangeDirective start={0} end={40}
color="#30B32D"></RangeDirective>
        <RangeDirective
          start={40}
          end={80}
          color="#FFDD00"
        ></RangeDirective>
        <RangeDirective
          start={80}

```

```

        end={120}
        color="#F03E3E"
      ></RangeDirective>
    </RangesDirective>
    <PointersDirective>
      <PointerDirective
        value={65}
        radius="60%"
        color="#757575"
        pointerWidth={8}
        animation={{
          enable: false,
        }}
        cap={{
          radius: 7,
          color: '#757575',
        }}
        needleTail={{
          length: '18%',
        }}
      ></PointerDirective>
    </PointersDirective>
  </AxisDirective>
</AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import * as React from "react";
import * as ReactDOM from "react-dom";
import { CircularGaugeComponent, AxesDirective, AxisDirective,
  PointersDirective, PointerDirective, RangesDirective, RangeDirective,
  GaugeTooltip, Legend, Inject } from '@syncfusion/ej2-react-circulargauge';
export function App() {
  return (
    <CircularGaugeComponent
      enableRtl="true"
      tooltip={{
        type: ['Pointer', 'Range'],
        format: 'Pointer : {value} ',
        enable: true,
        enableAnimation: false,
      }}
      legendSettings={{
        visible: true,
      }}
    >
      <Inject services={[GaugeTooltip, Legend]} />
      <AxesDirective>
        <AxisDirective
          minimum={0}

```

```

maximum={120}
startAngle={210}
endAngle={150}
radius="80%"
direction="AntiClockWise"
majorTicks={{
  height: 10,
  offset: 5,
  color: '#9E9E9E',
}}
minorTicks={{
  height: 0,
}}
lineStyle={{ width: 10, color: 'transparent' }}
labelStyle={{
  position: 'Inside',
  useRangeColor: false,
  font: {
    size: '12px',
    color: '#424242',
    fontFamily: 'Roboto',
    fontStyle: 'Regular',
  },
}}
>
  <RangesDirective>
    <RangeDirective start={0} end={40}
color="#30B32D"></RangeDirective>
    <RangeDirective
      start={40}
      end={80}
      color="#FFDD00"
    ></RangeDirective>
    <RangeDirective
      start={80}
      end={120}
      color="#F03E3E"
    ></RangeDirective>
  </RangesDirective>
  <PointersDirective>
    <PointerDirective
      value={65}
      radius="60%"
      color="#757575"
      pointerWidth={8}
      animation={{
        enable: false,
      }}
      cap={{
        radius: 7,
        color: '#757575',
      }}
      needleTail={{
        length: '18%',
      }}
    ></PointerDirective>
  </PointersDirective>

```

```

    </AxisDirective>
  </AxesDirective>
</CircularGaugeComponent>);
}
const root = ReactDOM.createRoot(document.getElementById('container'));
root.render(<App />);
{% endraw %}

```

Ej1 api migration in React Circular gauge component

This article describes the API migration process of Accordion component from Essential JS 1 to Essential JS 2.

Circular gauge dimensions

{% raw %}

| **Behavior** | **API in Essential JS 1** | **API in Essential JS 2** |

| --- | --- | --- |

| Height | **Property:** *height*

 <EJ.CircularGauge id="circulargauge" height={500}></EJ.CircularGauge>,

 document.getElementById('circulargauge') | **Property:** *height*

 <CircularGaugeComponent id='circulargauge' height="650"></CircularGaugeComponent>,

 document.getElementById('circulargauge'); |

| Width | **Property:** *width*

 <EJ.CircularGauge id="circulargauge" width={500}></EJ.CircularGauge>,

 document.getElementById('circulargauge') | **Property:** *width*

 <CircularGaugeComponent id='circulargauge' width="80%"></CircularGaugeComponent>,

 document.getElementById('circulargauge'); |

| Height(In Percentage) | Not Applicable | **Property:** *height*

 <CircularGaugeComponent id='circulargauge' height='100%'></CircularGaugeComponent>,

 document.getElementById('circulargauge'); |

| Width(In Percentage) | Not Applicable | **Property:** *width*

 <CircularGaugeComponent id='circulargauge' width="100%"></CircularGaugeComponent>,

 document.getElementById('circulargauge'); |

Axis Line

| **Behavior** | **API in Essential JS 1** | **API in Essential JS 2** |

| --- | --- | --- |

| Axisline Width | **Property:** *scales.size*

 <EJ.CircularGauge id="circulargauge" scales = {scales}></EJ.CircularGauge>,
 document.getElementById('circulargauge')

 var scales = [{ showScaleBar: true, size: 6}] | **Property:** *axes.lineStyle.width*

 <CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective lineStyle = {{width:

```
2}}></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document.getElementBy
```

```
Id('circulargauge'));|  
|Axisline Color| Property: scales.color<br/><br/> <EJ.CircularGauge id="circulargauge" scales =  
{scales}></EJ.CircularGauge>,<br/>document.getElementById('circulargauge')<br/><br/>var  
scales =[{ showScaleBar: true, color:"red"}]| Property: axes.lineStyle.width<br/><br/>  
<CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective lineStyle = {{color:  
'red'  
}}></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document.getElementById('circulargauge')<br/><br/>|
```

```
|Axisline BackgroundColor| Not Applicable| Property: axes.background<br/><br/>  
<CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective  
background='red'></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document.getElementById('circulargauge')<br/><br/>|
```

```
|Axisline Direction| Property: scales.direction<br/><br/> <EJ.CircularGauge id="circulargauge"  
scales =  
{scales}></EJ.CircularGauge>,<br/>document.getElementById('circulargauge')<br/><br/>var  
scales =[{ direction: "counterclockwise"}]| Property: axes.direction<br/><br/>  
<CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective direction=  
'AntiClockWise'></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document  
.getElementById('circulargauge')<br/><br/>|
```

```
|Axisline Radius| Property: scales.radius<br/><br/> <EJ.CircularGauge id="circulargauge" scales =  
{scales}></EJ.CircularGauge>,<br/>document.getElementById('circulargauge')<br/><br/>var  
scales =[{ showScaleBar: true, radius: 150}]| Property: axes.radius<br/><br/>  
<CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective  
radius='150'></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document.get  
ElementById('circulargauge')<br/><br/>|
```

```
|Axisline Startangle| Property: scales.startAngle<br/><br/> <EJ.CircularGauge id="circulargauge"  
scales =  
{scales}></EJ.CircularGauge>,<br/>document.getElementById('circulargauge')<br/><br/>var  
scales =[{ startAngle: 80}]| Property: axes.startAngle<br/><br/> <CircularGaugeComponent  
id='circulargauge'><AxesDirective><AxisDirective startAngle= {200}  
></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document.getElementBy  
Id('circulargauge')<br/><br/>|
```

```
|Axisline Endangle| Property: scales.sweepAngle<br/><br/> <EJ.CircularGauge id="circulargauge"  
scales =  
{scales}></EJ.CircularGauge>,<br/>document.getElementById('circulargauge')<br/><br/>var  
scales =[{ sweepAngle: 250}]| Property: axes.endAngle<br/><br/> <CircularGaugeComponent  
id='circulargauge'><AxesDirective><AxisDirective endAngle={150}  
></AxisDirective></AxesDirective>  
</CircularGaugeComponent>,<br/>document.getElementById('circulargauge')<br/><br/>|
```

```
| Minimum Axisvalue | Property: scales.minimum<br><br> <EJ.CircularGauge id="circulargauge"
scales =
{scales}></EJ.CircularGauge>,<br>document.getElementById('circulargauge')<br><br>var
scales =[{ minimum: 20 }]| Property: axes.minimum<br><br> <CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective
minimum={200}></AxisDirective></AxesDirective>
</CircularGaugeComponent>,<br>document.getElementById('circulargauge'));|

| Maximum Axisvalue | Property: scales.maximum<br><br> <EJ.CircularGauge id="circulargauge"
scales =
{scales}></EJ.CircularGauge>,<br>document.getElementById('circulargauge')<br><br>var
scales =[{ maximum: 200}]| Property: axes.maximum<br><br> <CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective
maximum={200}></AxisDirective></AxesDirective>
</CircularGaugeComponent>,<br>document.getElementById('circulargauge'));|
```

Ticks

| **Behavior** | **API in Essential JS 1** | **API in Essential JS 2** |

| --- | --- | --- |

```
| Type of Ticks | Property: scales.ticks.type<br><br> <EJ.CircularGauge id="circulargauge"
scales={scales}></EJ.CircularGauge>,<br>document.getElementById('circulargauge')<br>var
scales =[{ticks:[{ type: "major"}]}]| Property: axes.majorTicks<br><br>
<CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective majorTicks =
{{width:
2}}></AxisDirective></AxesDirective></CircularGaugeComponent>,<br>document.getElement
ById('circulargauge'));|
```

```
| Height of Major Ticks | Property: scales.ticks.height<br><br> <EJ.CircularGauge id="circulargauge"
scales={scales}></EJ.CircularGauge>,<br>document.getElementById('circulargauge')<br>var
scales =[{ticks:[{ type: "major", height: 12}]}]| Property: axes.majorTicks.height<br><br>
<CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective majorTicks =
{{height:
12}}></AxisDirective></AxesDirective></CircularGaugeComponent>,<br>document.getElement
ById('circulargauge'));|
```

```
| Width of Major Ticks | Property: scales.ticks.width<br><br> <EJ.CircularGauge id="circulargauge"
scales={scales}></EJ.CircularGauge>,<br>document.getElementById('circulargauge')<br>var
scales =[{ticks:[{ type: "major", width: 3}]}]| Property: axes.majorTicks.width<br><br>
<CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective majorTicks =
{{width:
3}}></AxisDirective></AxesDirective></CircularGaugeComponent>,<br>document.getElement
ById('circulargauge'));|
```

```
| Color of Major Ticks | Property: scales.ticks.color<br><br> <EJ.CircularGauge id="circulargauge"
scales={scales}></EJ.CircularGauge>,<br>document.getElementById('circulargauge')<br>var
scales =[{ticks:[{ type: "major",color: "#777777"}]}]| Property: axes.majorTicks.color<br><br>
```



```
<CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective majorTicks =
{{color: "#777777"
}}></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document.getElementById('circulargauge'));
```

| Offset of Major Ticks | **Property:** *scales.ticks.distanceFromScale*

 <EJ.CircularGauge id="circulargauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('circulargauge')
var
scales = [{ticks: [{ type: "major", distanceFromScale: 10 }]}] | **Property:**
axes.majorTicks.offset

 <CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective majorTicks = {{offset:
10}}></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElement
ById('circulargauge'));

| Angle of Major Ticks | **Property:** *scales.ticks.angle*

 <EJ.CircularGauge id="circulargauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('circulargauge')
var
scales = [{ticks: [{ type: "major", angle: 10 }]}] | Not Applicable |

| Interval of Major Ticks | **Property:** *scales.majorIntervalValue*

 <EJ.CircularGauge
id="circulargauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('circulargauge')
var
scales = [{ticks: [{ type: "major", majorIntervalValue: 10 }]}] | **Property:**
axes.majorTicks.interval

 <CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective majorTicks = {{interval:
10}}></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElement
ById('circulargauge'));

| Height of Minor Ticks | **Property:** *scales.ticks.height*

 <EJ.CircularGauge id="circulargauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('circulargauge')
var
scales = [{ticks: [{ type: "minor", height: 12 }]}] | **Property:** *axes.minorTicks.height*

<CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective minorTicks = {{
height:
12}}></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElement
ById('circulargauge'));

| Width of Minor Ticks | **Property:** *scales.ticks.width*

 <EJ.CircularGauge id="circulargauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('circulargauge')
var
scales = [{ticks: [{ type: "minor", width: 3 }]}] | **Property:** *axes.minorTicks.width*

<CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective minorTicks =
{{width:
3}}></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElement
ById('circulargauge'));

| Color of Minor Ticks | **Property:** *scales.ticks.color*

 <EJ.CircularGauge id="circulargauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('circulargauge')
var
scales = [{ticks: [{ type: "minor", color: "#777777" }]}] | **Property:** *axes.minorTicks.color*

<CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective minorTicks = {{

```
color: "#777777"
}}></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document.getElementById('circulargauge'));|
```

| Offset of Minor Ticks | **Property:** *scales.ticks.distanceFromScale*

 <EJ.CircularGauge id="circulargauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('circulargauge')
var
scales = [{ticks: [{ type: "minor", distanceFromScale: 10}]]} | **Property:**
axes.minorTicks.offset

 <CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective minorTicks = {{offset:
10}}></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElement
ById('circulargauge'));|

| Angle of Major Ticks | **Property:** *scales.ticks.angle*

 <EJ.CircularGauge id="circulargauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('circulargauge')
var
scales = [{ticks: [{ type: "minor", angle: 10 }]]} | Not Applicable |

| Interval of Minor Ticks | **Property:** *scales.majorIntervalValue*

 <EJ.CircularGauge
id="circulargauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('circulargauge')
var
scales = [{ticks: [{ type: "minor", majorIntervalValue: 10}]]} | **Property:**
axes.minorTicks.interval

 <CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective minorTicks = {{interval:
10}}></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElement
ById('circulargauge'));|

Labels

| **Behavior** | **API in Essential JS 1** | **API in Essential JS 2** |

| --- | --- | --- |

| Autoangle | **Property:** *scales.labels.autoAngle*

 <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales
= [{labels: [{ showLabels: true, autoAngle: true}]]} | **Property:** *axes.labelStyle.autoAngle*

<CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective labelStyle =
{{autoAngle:
true}}></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getEleme
ntById('circulargauge'));|

| Angle | **Property:** *scales.labels.angle*

 <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales
= [{labels: [{ showLabels: true, angle: 30}]]} | Not Applicable |

| Offset | **Property:** *scales.labels.distanceFromScales*

 <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales
= [{labels: [{ showLabels: true, distanceFromScales: 10 }]]} | **Property:**
axes.labelStyle.offset

 <CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective labelStyle = {{offset:

```
5}}></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document.getElementBy
```

```
ById('circulargauge'));|
| Format | Property: scales.labels.unitText<br/><br/> <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,<br/>document.getElementById('gauge')<br/>var scales
=[{labels: [{ unitText: "kmph", unitTextPosition: "front" }]}] | Property:
axes.labelStyle.format<br/><br/> <CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective labelStyle = {{format:
"kmph"}}></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document.getEle
```

```
mentById('circulargauge'));|
| UnitText Position | Property: scales.labels.placement<br/><br/> <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,<br/>document.getElementById('gauge')<br/>var scales
=[{labels: [{ showLabels: true, placement: "near" }]}] | Property: axes.labelStyle.position<br/><br/>
<CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective labelStyle =
{{position: "Outside"
}}></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document.getElementBy
```

```
Id('circulargauge'));|
| Label Range Color | Not Applicable | Property: axes.labelStyle.useRangeColor<br/><br/>
<CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective labelStyle =
{{useRangeColor:
true}}></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document.getEleme
```

```
ntById('circulargauge'));|
| LabelText Color | Property: scales.labels.color<br/><br/> <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,<br/>document.getElementById('gauge')<br/>var scales
=[{labels: [{color: "red" }]}] | Property: axes.labelStyle.font.color<br/><br/>
<CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective labelStyle = {{font:
{ color: "red" }
}}></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document.getElementBy
```

```
Id('circulargauge'));|
| Opacity | Property: scales.labels.opacity<br/><br/> <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,<br/>document.getElementById('gauge')<br/>var scales
=[{labels: [{ opacity: 0.3 }]}] | Property: axes.labelStyle.font.opacity<br/><br/>
<CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective labelStyle = {{font:
{ opacity: 0.5
}}></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document.getElementB
```

```
ylId('circulargauge'));|
| Label Font Family | Property: scales.labels.font.fontFamily<br/><br/> <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,<br/>document.getElementById('gauge')<br/>var scales
=[{labels: [{ font: { fontFamily: "Arial" } }]}] | Property: axes.labelStyle.font.fontFamily<br/><br/>
<CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective labelStyle = {{font:
{ fontFamily: 'Roboto'

```

```
}}></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document.getElementById('circulargauge'));|
```

| Label Font Style | **Property:** *scales.labels.font.fontStyle*

 <EJ.CircularGauge id="gauge" scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales =[{labels: [{font: { fontStyle: "Bold" } }]}]| **Property:** *axes.labelStyle.font.fontStyle*

 <CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective labelStyle = {{font: { fontStyle: 'Bold' } }}></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElementById('circulargauge'));|

| Label Font Size | **Property:** *scales.labels.font.size*

 <EJ.CircularGauge id="gauge" scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales =[{labels: [{ font: { size: "12px" } }]}]| **Property:** *axes.labelStyle.font.size*

 <CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective labelStyle = {{font: { size: '12px' } }}></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElementById('circulargauge'));|

| Label Font Weight | Not Applicable | **Property:** *axes.labelStyle.font.fontWeight*

 <CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective labelStyle = {{font: { fontWeight: 'Regular' } }}></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElementById('circulargauge'));|

Ranges

| **Behavior** | **API in Essential JS 1** | **API in Essential JS 2** |

| --- | --- | --- |

| Start Value | **Property:** *scales.ranges.startValue*

 <EJ.CircularGauge id="gauge" scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales =[{showRanges: true,ranges: [{ startValue: 20}]}]| **Property:** *axes.ranges.start*

 <CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective><RangesDirective><RangeDirective start={20}></RangeDirective></RangesDirective></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElementById('circulargauge'));|

| End Value | **Property:** *scales.ranges.endValue*

 <EJ.CircularGauge id="gauge" scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales =[{showRanges: true,ranges: [{endValue: 30 }]}]| **Property:** *axes.ranges.end*

 <CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective><RangesDirective><RangeDirective end={30}></RangeDirective></RangesDirective></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElementById('circulargauge'));|

| Start Width | **Property:** *scales.ranges.startWidth*

 <EJ.CircularGauge id="gauge" scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales =[{showRanges: true,ranges: [{ startWidth: 10}]}]| **Property:** *axes.ranges.startWidth*


```

<CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><RangesDirective><RangeDirective
startWidth={10}
></RangeDirective></RangesDirective></AxisDirective></AxesDirective></CircularGaugeComp
onent>,<br/>document.getElementById('circulargauge'));|

|End Width| Property: scales.ranges.endWidth<br/><br/> <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,<br/>document.getElementById('gauge')<br/>var scales
=[{showRanges: true,ranges: [{endWidth: 10 }]}]| Property: axes.ranges.endWidth<br/><br/>
<CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><RangesDirective><RangeDirective
endWidth={10}></RangeDirective></RangesDirective></AxisDirective></AxesDirective></Circul
arGaugeComponent>,<br/>document.getElementById('circulargauge'));|

|Color| Property: scales.ranges.backgroundColor<br/><br/> <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,<br/>document.getElementById('gauge')<br/>var scales
=[{showRanges: true,ranges: [{ backgroundColor: "red"}]}]| Property: axes.ranges.color<br/><br/>
<CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><RangesDirective><RangeDirective color=
"red"></RangeDirective></RangesDirective></AxisDirective></AxesDirective></CircularGaugeC
omponent>,<br/>document.getElementById('circulargauge'));|

|Offset| Property: scales.ranges.distanceFromScale<br/><br/> <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,<br/>document.getElementById('gauge')<br/>var scales
=[{showRanges: true,ranges: [{distanceFromScale: 10 }]}]| Not Applicable|

|Placement| Property: scales.ranges.placement<br/><br/> <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,<br/>document.getElementById('gauge')<br/>var scales
=[{showRanges: true,ranges: [{ placement: "center"}]}]| Not Applicable|

|Opacity| Property: scales.ranges.opacity<br/><br/> <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,<br/>document.getElementById('gauge')<br/>var scales
=[{showRanges: true,ranges: [{ opacity: 0.5}]}]| Not Applicable|

|Radius| Not Applicable| Property: axes.ranges.radius<br/><br/> <CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><RangesDirective><RangeDirective radius=
'80'></RangeDirective></RangesDirective></AxisDirective></AxesDirective></CircularGaugeCo
mponent>,<br/>document.getElementById('circulargauge'));|

|Rounded Corner Radius| Not Applicable| Property: axes.ranges.roundedCornerRadius<br/><br/>
<CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><RangesDirective><RangeDirective
roundedCornerRadius={10}></RangeDirective></RangesDirective></AxisDirective></AxesDirect
ive></CircularGaugeComponent>,<br/>document.getElementById('circulargauge'));|

|Gradients| Property: scales.ranges.gradients<br/><br/> <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,<br/>document.getElementById('gauge')<br/>var scales
=[{showRanges: true,ranges: [{ showRanges: true, radiants: { colorInfo: [{ colorStop : 0,
color:"#FFFFFF" } ] }]}]| Not Applicable|

```

|Border| **Property:** *scales.ranges.border*

 <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales
=[{showRanges: true,ranges: [{ border: { color: "blue", width: 2 } }]}]| Not Applicable|

Needle Pointer

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

|Needle Pointer| **Property:** *scales.pointers.type*

 <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales
=[{pointers: [{ type: 'needle' }]}]| **Property:** *axes.pointers.type*

<CircularGaugeComponent id='circulargauge'><
AxesDirective><AxisDirective><PointersDirective><PointerDirective type= 'needle'
value={20}></PointerDirective></PointersDirective>
</AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElementById(
'circulargauge'))|

|Needle Pointer Color| **Property:** *scales.pointers.backgroundColor*

 <EJ.CircularGauge
id="gauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales
=[{pointers: [{ type: 'needle',backgroundColor: 'red' }]}]| **Property:**
axes.pointers.color

<CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><PointersDirective><PointerDirective type=
'needle' color=
'red'></PointerDirective></PointersDirective></AxisDirective></AxesDirective></CircularGauge
Component>,
document.getElementById('circulargauge'))|

|Animation| **Property:** *enableAnimation*

 <EJ.CircularGauge id="gauge"
enableAnimation={true}></EJ.CircularGauge>,
document.getElementById('gauge')|
Property: *axes.pointers.animation*

 <CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><PointersDirective><PointerDirective type=
'needle' animation: true duration= {1000}
></PointerDirective></PointersDirective></AxisDirective></AxesDirective></CircularGaugeCom
ponent>,
document.getElementById('circulargauge'))|

|Pointer Width| **Property:** *scales.pointers.width*

 <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales
=[{pointers: [{ type: 'needle',width: 5 }]}]| **Property:** *axes.pointers.pointerWidth*

<CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><PointersDirective><PointerDirective type=
'needle' pointerWidth= {5}
></PointerDirective></PointersDirective></AxisDirective></AxesDirective></CircularGaugeCom
ponent>,
document.getElementById('circulargauge'))|

|Pointer Radius| **Property:** *scales.pointers.distanceFromScale*

 <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales
=[{pointers: [{ type: 'needle', distanceFromScale: 10 }]}]| **Property:**

```
axes.pointers.radius<br/><br/><CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><PointersDirective><PointerDirective
radius={80}></PointerDirective></PointersDirective></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document.getElementById('circulargauge')));|
```

| Opacity | **Property:** *scales.pointers.opacity*

 <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales
=[{pointers: [{ type: 'needle',opacity: 0.5 }]}] | Not Applicable|

| Needle Type | **Property:** *scales.pointers.needleType*

 <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales
=[{pointers: [{ type: 'needle',needleType: "triangle" }]}] | Not Applicable|

| Back Needle Length | **Property:** *scales.pointers.backNeedleLength*

<EJ.CircularGauge
id="gauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales
=[{pointers: [{ type: 'needle',showBackNeedle: true, backNeedleLength: 3 }]}] | **Property:**
axes.pointers.needleTail.length

 <CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><PointersDirective><PointerDirective
needleTail= { length: 5 }></PointerDirective></PointersDirective>
</AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElementById(
'circulargauge')));|

Marker Pointer

| **Behavior** | **API in Essential JS 1** | **API in Essential JS 2** |

| --- | --- | --- |

| Marker Pointer | **Property:** *scales.pointers.type*

 <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales
=[{pointers: [{ type: 'marker' }]}] | **Property:** *axes.pointers.type*

<CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><PointersDirective><PointerDirective type=
'marker' value={20}></PointerDirective></PointersDirective>
</AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElementById(
'circulargauge')));|

| Marker Type | **Property:** *scales.pointers.markerType*

 <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales
=[{pointers: [{ type: 'marker', markerType: "rectangle" }]}] | **Property:**
axes.pointers.markerShape

 <CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><PointersDirective><PointerDirective type=
'marker'
markerShape='Diamond'></PointerDirective></PointersDirective></AxisDirective></AxesDirect
ive></CircularGaugeComponent>,
document.getElementById('circulargauge')));|

| Marker Width | **Property:** *scales.pointers.width*

 <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales
=[{pointers: [{ type: 'marker', width: 20 }]}] | **Property:** *axes.pointers.markerWidth*


```
<CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><PointersDirective><PointerDirective type=
'marker' markerWidth= 20 ></PointerDirective>
</PointersDirective></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>docum
ent.getElementById('circulargauge'));|
```

| Marker Height| **Property:** *scales.pointers.length*

 <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales
=[{pointers: [{ type: 'marker', length: 25 }]}]| **Property:** *axes.pointers.markerHeight*

<CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><PointersDirective><PointerDirective type=
'marker' markerHeight={25}></PointerDirective>
</PointersDirective></AxisDirective></AxesDirective></CircularGaugeComponent>,
docum
ent.getElementById('circulargauge'));|

| Marker Image| **Property:** *scales.pointers.imageUrl*

 <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales
=[{pointers: [{ type: 'marker', imageUrl: "football.png" }]}]| **Property:**
axes.pointers.imageUrl

 <CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><PointersDirective><PointerDirective type=
'marker' imageUrl= "football.png"></PointerDirective>
</PointersDirective></AxisDirective></AxesDirective></CircularGaugeComponent>,
docum
ent.getElementById('circulargauge'));|

| Border Customization| **Property:** *scales.pointers.border*

 <EJ.CircularGauge id="gauge"
scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales
=[{pointers: [{ type: 'marker', border: { color: 'red', width: 2 } }]}]| **Property:**
axes.pointers.border

 <CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><PointersDirective><PointerDirective type=
'marker' border={ color: 'red', width: 2 }></PointerDirective>
</PointersDirective></AxisDirective></AxesDirective></CircularGaugeComponent>,
docum
ent.getElementById('circulargauge'));|

Rangebar Pointer

| **Behavior** | **API in Essential JS 1** | **API in Essential JS 2** |

| --- | --- | --- |

| Rangebar| Not Applicable| **Property:** *axes.pointers.type*

 <CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><PointersDirective><PointerDirective type=
'RangeBar' ></PointerDirective>
</PointersDirective></AxisDirective></AxesDirective></CircularGaugeComponent>,
docum
ent.getElementById('circulargauge'));|

| Rounded Corner Radius| Not Applicable| **Property:** *axes.pointers.roundedCornerRadius*

<CircularGaugeComponent
id='circulargauge'><AxesDirective><AxisDirective><PointersDirective><PointerDirective type=
'RangeBar' roundedCornerRadius={10}></PointerDirective>


```
</PointersDirective></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document.getElementById('circulargauge')));|
```

Annotations

| **Behavior** | **API in Essential JS 1** | **API in Essential JS 2** |

| --- | --- | --- |

|Content| **Property:** *scales.customLabels.value*

<EJ.CircularGauge id="gauge" scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales = [{ showCustomLabels: true, customLabels: [{ value: 'Lineargauge' }]}] | **Property:** *axes.annotations.content*

 <CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective><AnnotationsDirective><AnnotationDirective content= 'Annotation' ></AnnotationDirective></AnnotationsDirective></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElementById('circulargauge')));|

|Angle| **Property:** *scales.customLabels.textAngle*

<EJ.CircularGauge id="gauge" scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales = [{ showCustomLabels: true, customLabels: [{ value: 'Lineargauge', textAngle: 90}]}] | **Property:** *axes.annotations.angle*

 <CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective><AnnotationsDirective><AnnotationDirective content= 'Annotation' angle={90}></AnnotationDirective></AnnotationsDirective></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElementById('circulargauge')));|

|Font Family| **Property:** *scales.customLabels.font.fontFamily*

<EJ.CircularGauge id="gauge" scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales = [{ showCustomLabels: true, customLabels: [{ value: 'Lineargauge', font: { fontFamily: "Arial" } }]}] | **Property:** *axes.annotations.textStyle.fontFamily*

 <CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective><AnnotationsDirective><AnnotationDirective content= 'Annotation' textStyle= { fontFamily: "Arial" }></AnnotationDirective></AnnotationsDirective></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElementById('circulargauge')));|

|Font Color| **Property:** *scales.customLabels.color*

<EJ.CircularGauge id="gauge" scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales = [{ showCustomLabels: true, customLabels: [{ value: 'Lineargauge', color : "red"}]]] | **Property:** *axes.annotations.textStyle.color*

 <CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective><AnnotationsDirective><AnnotationDirective content= 'Annotation' textStyle= { color: "red" }></AnnotationDirective></AnnotationsDirective></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElementById('circulargauge')));|

|Auto Angle| Not Applicable| **Property:** *axes.annotations.autoAngle*

 <CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective><AnnotationsDirective><AnnotationDirective content= 'Annotation' autoAngle = true

```
></AnnotationDirective></AnnotationsDirective></AxisDirective></AxesDirective></CircularGaugeComponent>,<br/>document.getElementById('circulargauge')));|
```

| Radius | Not Applicable | **Property:** *axes.annotations.radius*

 <CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective><AnnotationsDirective><AnnotationDirective content= 'Annotation' radius = "10%"></AnnotationDirective></AnnotationsDirective></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElementById('circulargauge')));|

| Annotation Position | **Property:** *scales.customLabels.position*

 <EJ.CircularGauge id="gauge" scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales = [{ showCustomLabels: true, customLabels: [{ value: 'Lineargauge', position : { x: 10, y: 10 } }]}]| Not Applicable|

| Annotation Position Type | **Property:** *scales.customLabels.positionType*

 <EJ.CircularGauge id="gauge" scales={scales}></EJ.CircularGauge>,
document.getElementById('gauge')
var scales = [{ showCustomLabels: true, customLabels: [{ value: 'Lineargauge',positionType : "outer" }]}]| Not Applicable|

| ZIndex | Not Applicable | **Property:** *axes.annotations.zIndex*

 <CircularGaugeComponent id='circulargauge'><AxesDirective><AxisDirective><AnnotationsDirective><AnnotationDirective content= 'Annotation' zIndex = '1'></AnnotationDirective></AnnotationsDirective></AxisDirective></AxesDirective></CircularGaugeComponent>,
document.getElementById('circulargauge')));|

Appearance

| **Behavior** | **API in Essential JS 1** | **API in Essential JS 2** |

| --- | --- | --- |

| Title | Not Applicable | **Property:** *title*

 <CircularGaugeComponent id='circulargauge'title= 'Circular Gauge'></CircularGaugeComponent>,
document.getElementById('circulargauge')));|

| Background Color | **Property:** *backgroundColor*

 <EJ.CircularGauge id="gauge" backgroundColor = "red" ></EJ.CircularGauge>,
document.getElementById('gauge')| **Property:** *background*

 <CircularGaugeComponent id='circulargauge' background = "red" ></CircularGaugeComponent>,
document.getElementById('circulargauge')));|

| Localization | **Property:** *locale*

 <EJ.CircularGauge id="gauge" locale = "en-US"></EJ.CircularGauge>,
document.getElementById('gauge')| **Property:** *locale*

 <CircularGaugeComponent id='circulargauge' locale = "en-US"></CircularGaugeComponent>,
document.getElementById('circulargauge')));|

| Border | Not Applicable | **Property:** *border*

 <CircularGaugeComponent id='circulargauge' border ={ color: "red" , width: 2 }></CircularGaugeComponent>,
document.getElementById('circulargauge')));|

|Center of X| Not Applicable| **Property:** *centerX*

 <CircularGaugeComponent id='circulargauge' centerX = "120px"></CircularGaugeComponent>,
document.getElementById('circulargauge'));|

|Center of Y| Not Applicable| **Property:** *centerY*

 <CircularGaugeComponent id='circulargauge' centerY = "150px"></CircularGaugeComponent>,
document.getElementById('circulargauge'));|

|Theme| **Property:** *theme*

<EJ.CircularGauge id="gauge" theme = "flatlight"></EJ.CircularGauge>,
document.getElementById('gauge')| **Property:** *theme*

<CircularGaugeComponent id='circulargauge' theme = "Material"></CircularGaugeComponent>,
document.getElementById('circulargauge'));|

|Margin| Not Applicable| **Property:** *margin*

 <CircularGaugeComponent id='circulargauge' margin= { left: 40, right: 40, top: 40, bottom: 40 }></CircularGaugeComponent>,
document.getElementById('circulargauge'));|

Events

| **Behavior** | **API in Essential JS 1** | **API in Essential JS 2** |

| --- | --- | --- |

|Annotation Event| **Event:** *drawCustomLabel*

 <EJ.CircularGauge id="gauge" drawCustomLabel={drawCustomLabel}></EJ.CircularGauge>,
document.getElementById('gauge')
function drawCustomLabel(args) {}| **Event:** *annotationRender*

 <CircularGaugeComponent id='circulargauge' annotationRender={this.annotationRender.bind(this)}></CircularGaugeComponent>,
document.getElementById('circulargauge'));
public annotationRender(args: IAnnotationRenderEventArgs): void {}|

|Label Event| **Event:** *drawLabels*

 <EJ.CircularGauge id="gauge" drawLabels={drawLabels}></EJ.CircularGauge>,
document.getElementById('gauge')
function drawLabels(args) {}| **Event:** *axisLabelRender*

 <CircularGaugeComponent id='circulargauge' axisLabelRender={this.axisLabelRender.bind(this)}></CircularGaugeComponent>,
document.getElementById('circulargauge'));
public axisLabelRender(args: IAxisLabelRenderEventArgs): void {}|

|Load Event| **Event:** *load*

 <EJ.CircularGauge id="gauge" load={load}></EJ.CircularGauge>,
document.getElementById('gauge')
function load(args) {}| **Event:** *load*

 <CircularGaugeComponent id='circulargauge' load={this.load.bind(this)}></CircularGaugeComponent>,
document.getElementById('circulargauge'));
public load(args: ILoadedEventArgs): void {}|

|Loaded Event| **Event:** *loaded*

 <EJ.CircularGauge id="gauge" loaded={loaded}></EJ.CircularGauge>,
document.getElementById('gauge')
function loaded(args) {}| **Event:** *loaded*

 <CircularGaugeComponent id='circulargauge' loaded={this.loaded.bind(this)}></CircularGaugeComponent>,
document.getElementById('circulargauge'));
public loaded(args: ILoadedEventArgs): void {}|

| Tooltip Rendered Event | Not Applicable | **Event:** *tooltipRender*

 <CircularGaugeComponent id='circulargauge'
 tooltipRender={this.tooltipRender.bind(this)}></CircularGaugeComponent>,
document.get
 ElementById('circulargauge'));
public tooltipRender(args: ITooltipRenderEventArgs): void
 {}|

| Resized Rendered Event | Not Applicable | **Event:** *resized*

 <CircularGaugeComponent
 id='circulargauge'
 tooltipRender={this.tooltipRender.bind(this)}></CircularGaugeComponent>,
document.get
 ElementById('circulargauge'));
public tooltipRender(args: IResizeEventArgs): void {}|

| Animation Event | Not Applicable | **Event:** *animationComplete*

 <CircularGaugeComponent
 id='circulargauge'
 animationComplete={this.animationComplete.bind(this)}></CircularGaugeComponent>,
do
 cument.getElementById('circulargauge'));
public animationComplete(args:
 IAnimationCompleteEventArgs): void {}|

| Mousedown Event | **Event:** *mouseClick*

 <EJ.CircularGauge id="gauge" mouseClick={
 mouseClick}></EJ.CircularGauge>,
document.getElementById('gauge')
function
 mouseClick(args) {}| **Event:** *gaugeMouseDown*

 <CircularGaugeComponent
 id='circulargauge'
 gaugeMouseDown={this.gaugeMouseDown.bind(this)}></CircularGaugeComponent>,
docu
 ment.getElementById('circulargauge'));
public gaugeMouseDown(args: IMouseEventArgs):
 void {}|

| Mousemove Event | **Event:** *mouseClickMove*

 <EJ.CircularGauge id="gauge"
 mouseClickMove={mouseClickMove}></EJ.CircularGauge>,
document.getElementById('gau
 ge')
function mouseClickMove(args) {}| **Event:**
gaugeMouseLeave

 <CircularGaugeComponent id='circulargauge'
 gaugeMouseLeave={this.gaugeMouseLeave.bind(this)}></CircularGaugeComponent>,
docu
 ment.getElementById('circulargauge'));
public gaugeMouseLeave(args: IMouseEventArgs):
 void {}|

| Mouseup Event | **Event:** *mouseClickUp*

 <EJ.CircularGauge id="gauge" mouseClickUp={
 mouseClickUp}></EJ.CircularGauge>,
document.getElementById('gauge')
function
 mouseClickUp(args) {}| **Event:** *gaugeMouseUp*

 <CircularGaugeComponent
 id='circulargauge'
 gaugeMouseUp={this.gaugeMouseUp.bind(this)}></CircularGaugeComponent>,
document.
 getElementById('circulargauge'));
public gaugeMouseUp(args: IMouseEventArgs): void {}|

| Pointerdrag Move Event | **Event:** *drawPointers*

 <EJ.CircularGauge id="gauge"
 drawpointers={drawpointers}></EJ.CircularGauge>,
document.getElementById('gauge')
function drawpointers(args) {}| **Event:** *dragMove*

 <CircularGaugeComponent
 id='circulargauge' dragMove={this.
 dragMove.bind(this)}></CircularGaugeComponent>,
document.getElementById('circularga
 uge'));
public dragMove(args: IMouseEventArgs): void {}|

| Draw Range Event | **Event:** *drawRange*
`<EJ.CircularGauge id="gauge" drawRange={drawRange}></EJ.CircularGauge>`,
`document.getElementById('gauge')`
 nction *drawRange*(args) {} | Not Applicable |

| Draw Ticks Event | **Event:** *drawTicks*
`<EJ.CircularGauge id="gauge" drawTicks={drawTicks}></EJ.CircularGauge>`,
`document.getElementById('gauge')`
 ion *drawTicks*(args) {} | Not Applicable |

| Legend Render Event | **Event:** *legendItemRender*
`<EJ.CircularGauge id="gauge" legendItemRender={legendItemRender}></EJ.CircularGauge>`,
`document.getElementById('gauge')`
 function *legendItemRender*(args) {} | Not Applicable |

| Right Click Event | **Event:** *rightClick*
`<EJ.CircularGauge id="gauge" drawpointers={drawpointers}></EJ.CircularGauge>`,
`document.getElementById('gauge')`
`>function drawpointers(args) {}` | Not Applicable |

| Double Click Event | **Event:** *doubleClick*
`<EJ.CircularGauge id="gauge" drawpointers={drawpointers}></EJ.CircularGauge>`,
`document.getElementById('gauge')`
`>function drawpointers(args) {}` | Not Applicable |

{% endraw %}

ColorPicker

Getting Started

The following section explains the required steps to build the ColorPicker component with its basic usage in step-by-step procedure.

Dependencies

The following list of dependencies are required to use the ColorPicker component in your application.

```
`javascript
```

```
|-- @syncfusion/ej2-react-inputs
```

```
|-- @syncfusion/ej2-react-base
```

```
|-- @syncfusion/ej2-base
```

```
|-- @syncfusion/ej2-inputs
```

```
|-- @syncfusion/ej2-popups
```

```
|-- @syncfusion/ej2-splitbuttons
```

```
,
```

Installation and configuration

You can use [Create-react-app](#) to setup the applications.

To install `create-react-app` run the following command.

```
`bash
```

```
npm install -g create-react-app
```

`

To set-up a React application in TypeScript environment, run the following command.

```
`bash
npx create-react-app my-app --template typescript
cd my-app
npm start
`
```

To set-up a React application in JavaScript environment, run the following command.

```
`bash
npx create-react-app my-app
cd my-app
npm start
`
```

Adding Syncfusion packages

All the available Essential JS 2 packages are published in [npmjs.com](https://www.npmjs.com) public registry.

You can choose the component that you want to install. For this application, we are going to use ColorPicker component.

To install ColorPicker component, use the following command

```
`bash
npm install @syncfusion/ej2-react-inputs --save
`
```

Adding CSS reference

Import the ColorPicker component's required CSS references as follows in `src/App.css`.

```
`css
@import "../node_modules/@syncfusion/ej2-base/styles/material.css";
@import "../node_modules/@syncfusion/ej2-inputs/styles/material.css";
@import "../node_modules/@syncfusion/ej2-popups/styles/material.css";
@import "../node_modules/@syncfusion/ej2-splitbuttons/styles/material.css";
@import "../node_modules/@syncfusion/ej2-inputs/styles/material.css";
@import "../node_modules/@syncfusion/ej2-buttons/styles/material.css";
`
```

Adding ColorPicker to the application

Now, you can start adding ColorPicker component to the application. We have added ColorPicker component in `src/App.tsx` file using following code.

```
`ts
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
import './App.css';
// initializes ColorPicker component
function App() {
  return ( <div style={{marginTop: '150px'}}>
    <div id='container'>
      <div className='wrap'>
        <h4>Choose Color</h4>
        <ColorPickerComponent id='color-picker' />
      </div>
    </div>
  </div>
);
};
export default App;
```

Run the application

Use the `npm run start` command to run the application in the browser.

```
npm run start
```

The following example shows the default ColorPicker.

INDEX.JSX

```
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
// initializes ColorPicker component.
function App() {
  return (
    <div id='container'>
      <div className='wrap'>
        <h4>Choose Color</h4>
        <ColorPickerComponent id='color-picker' />
      </div>
    </div>
  );
}
```

```
;
export default App;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
// initializes ColorPicker component.
function App() {
    return (
        <div id='container'>
            <div className='wrap'>
                <h4>Choose Color</h4>
                <ColorPickerComponent id='color-picker' />
            </div>
        </div>
    );
};
export default App;
ReactDOM.render(<App />, document.getElementById('element'));
```

See Also

- [Set color value](#)
- [ColorPicker customization](#)

Mode and value in React Color picker component

Inline

By default, the ColorPicker will be rendered using SplitButton and open the pop-up to access the ColorPicker. To render the ColorPicker container alone and to access it directly, render it as inline. It can be achieved by setting the [inline](#) property to `true`.

The following sample shows the inline type rendering of ColorPicker.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
//To render component as inline.
function App() {
    return (<div id='container'>
        <div className='wrap'>
            <h4>Choose Color</h4>
            <ColorPickerComponent inline={true}
showButtons={false}></ColorPickerComponent>
        </div>
    </div>);
};
export default App;
ReactDOM.render(<App />, document.getElementById('element'));
```


INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
//To render component as inline.
function App() {
    return (
        <div id='container'>
            <div className='wrap'>
                <h4>Choose Color</h4>
                <ColorPickerComponent inline={true}
showButtons={false}></ColorPickerComponent>
            </div>
        </div>
    );
};
export default App;
ReactDOM.render(<App />, document.getElementById('element'));
```

> The `showButtons` property is disabled in this sample because the control buttons are not needed for inline type. To know about the control buttons functionality, refer to the [showButtons](#) sample.

Rendering palette at initial load

By default, the `Picker` area will be rendered at initial load. To render the Palette area while opening the ColorPicker pop-up, and specify the `mode` property as `Palette`.

In the following sample, it will render the `Palette` at initial load.

INDEX.JSX

```
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
//To render palette at initial load.
function App() {
    return (<div id='container'>
        <div className='wrap'>
            <h4>Choose Color</h4>
            <ColorPickerComponent mode="Palette"/>
        </div>
    </div>);
};
export default App;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
//To render palette at initial load.
function App() {
```

```

    return (
      <div id='container'>
        <div className='wrap'>
          <h4>Choose Color</h4>
          <ColorPickerComponent mode="Palette"/>
        </div>
      </div>
    );
  };
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

Color value

The [value](#) property can be used to specify the color value to the ColorPicker. It supports either **three** or **six** digit hex codes. To include **opacity**, set the color value as **four** or **eight** digit hex code.

In the following sample, the color value sets as **four** digit hex code, the last digit represents the **opacity** value.

INDEX.JSX

```

import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
//To set color value.
function App() {
  return (<div id='container'>
    <div className='wrap'>
      <h4>Choose Color</h4>
      <ColorPickerComponent value="035a"/>
    </div>
  </div>);
}
;
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
//To set color value.
function App() {
  return (
    <div id='container'>
      <div className='wrap'>
        <h4>Choose Color</h4>
        <ColorPickerComponent value="035a"/>
      </div>
    </div>
  );
};
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

> The [value](#) property supports hex code with or without # prefix.

See Also

- [How to render palette alone](#)
- [Custom palette](#)
- [No color support in palette](#)

Localization in React Color picker component

Localization

The **Localization** library allows you to localize default text content of the ColorPicker. The ColorPicker component has static text for control buttons (apply / cancel) and mode switcher that can be changed to other cultures (Arabic, Deutsch, French, etc.) by defining the [locale](#) value and translation object.

The following list of properties and its values are used in the ColorPicker.

Locale key words | Text

Apply | Apply

Cancel | Cancel

ModeSwitcher | Switch Mode

Loading translations

To load translation object in an application use **load** function of **L10n** class.

The below example demonstrates the ColorPicker in **Deutsch** culture.

INDEX.JSX

```
import { L10n } from '@syncfusion/ej2-base';
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
L10n.load({
  'de-DE': {
    'colorpicker': {
      'Apply': 'Anwenden',
      'Cancel': 'Abbrechen',
      'ModeSwitcher': 'Modus wechseln'
    }
  }
});
function App() {
  return (<div id='container'>
    <div className='wrap'>
      <h4>Choose Color</h4>
      <ColorPickerComponent locale='de-DE' />
    </div>
  </div>);
}
export default App;
```

```
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import { L10n } from '@syncfusion/ej2-base';
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
L10n.load({
  'de-DE': {
    'colorpicker': {
      'Apply': 'Anwenden',
      'Cancel': 'Abbrechen',
      'ModeSwitcher': 'Modus wechseln'
    }
  }
});
function App() {
  return (
    <div id='container'>
      <div className='wrap'>
        <h4>Choose Color</h4>
        <ColorPickerComponent locale='de-DE' />
      </div>
    </div>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));
```

Right to Left - RTL

ColorPicker component has **RTL** support. It helps to render the ColorPicker from right-to-left direction.

It improves the user experiences and accessibility for users who use right-to-left languages(Arabic, Farsi, Urdu, etc). This can be achieved by setting the [enableRtl](#) property to **true**.

The following example illustrates how to enable right-to-left support in ColorPicker component.

INDEX.JSX

```
import { L10n } from '@syncfusion/ej2-base';
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
L10n.load({
  'ar-AE': {
    'colorpicker': {
      'Apply': 'تطبيق',
      'Cancel': 'إلغاء',
      'ModeSwitcher': 'مفتاح كهربائي الوضع'
    }
  }
});
function App() {
  return (<div id='container'>
    <div className='wrap'>
```

```

        <h4>Choose Color</h4>
        <ColorPickerComponent enableRtl={true} locale='ar-AE' />
      </div>
    </div>);
  }
  export default App;
  ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { L10n } from '@syncfusion/ej2-base';
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
L10n.load({
  'ar-AE': {
    'colorpicker': {
      'Apply': 'تطبيق',
      'Cancel': 'إلغاء',
      'ModeSwitcher': 'مفتاح كهربائي الوضع'
    }
  }
});
function App() {
  return (
    <div id='container'>
      <div className='wrap'>
        <h4>Choose Color</h4>
        <ColorPickerComponent enableRtl={true} locale='ar-AE' />
      </div>
    </div>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

See Also

- [More information about localization](#)

Accessibility in React ColorPicker component

The ColorPicker component followed the accessibility guidelines and standards, including [ADA, Section 508](#), [WCAG 2.2](#) standards, and [WCAG roles](#) that are commonly used to evaluate accessibility.

The accessibility compliance for the ColorPicker component is outlined below.

| Accessibility Criteria | Compatibility |

| -- | -- |

| [WCAG 2.2](#) Support | |

| [Section 508](#) Support | |

```

| Screen Reader Support |  |

| Right-To-Left Support |  |

| Color Contrast |  |

| Mobile Device Support |  |

| Keyboard Navigation Support |  |

| Accessibility Checker Validation |  |

| Axe-core Accessibility Validation |  |

<style>
.post .post-content img {
display: inline-block;
margin: 0.5em 0;
}
</style>

<div> - All
features of the component meet the requirement.</div>

<div> - Some features of the component do not meet the requirement.</div>

<div> - The component does not meet the requirement.</div>

```

WAI-ARIA attributes

The ColorPicker component followed the [WAI-ARIA](#) patterns to meet the accessibility. The following ARIA attributes are used in the ColorPicker component:

Attributes	Purpose
role	Indicates the ColorPicker component as <code>color</code> and the tiles as <code>gridcell</code> in the color palette.
aria-label	Indicates the accessible name for the tiles.
aria-selected	Indicates the current selected state of the tile.
aria-haspopup	Indicates the availability of the popup element.
aria-expanded	Indicates whether the popup can be expanded or collapsed, as well as indicates whether its current state is expanded or collapsed.

| **aria-owns** | Identifies an elements in order to define a visual, functional, or contextual parent/child relationship between DOM elements where the DOM hierarchy cannot be used to represent the relationship. |

| **aria-disabled** | Indicates that the element is perceivable but disabled, so it is not editable or otherwise operable. |

Keyboard interaction

The ColorPicker component followed the [keyboard interaction](#) guideline, making it easy for people who use assistive technologies (AT) and those who completely rely on keyboard navigation. The following keyboard shortcuts are supported by the ColorPicker component.

| **Press** | **To do this** |

| --- | --- |

| **Up Arrow** | Moves the handler/tile up from the current position. |

| **Down Arrow** | Moves the handler/tile down from the current position. |

| **Left Arrow** | Moves the handler/tile left from the current position. |

| **Right Arrow** | Moves the handler/tile right from the current position. |

| **Enter** | Apply the selected color value. |

| **Tab** | To focus the next focusable element in the ColorPicker popup. |

Ensuring accessibility

The ColorPicker component's accessibility levels are ensured through an [accessibility-checker](#) and [axe-core](#) software tools during automated testing.

The accessibility compliance of the ColorPicker component is shown in the following sample. Open the [sample](#) in a new window to evaluate the accessibility of the ColorPicker component with accessibility tools.

See also

- [Accessibility in Syncfusion React components](#)

Style and appearance in React Color picker component

To modify the ColorPicker appearance, you need to override the default CSS of ColorPicker component. Please find the list of CSS classes and its corresponding section in ColorPicker component. Also, you have an option to create your own custom theme for the controls using our [Theme Studio](#).

CSS Class | Purpose of Class

|.e-custom-picker .e-container .e-handler|To customized Color Picker selection handler

|.color-picker.e-dropdown-popup ul .e-container|To customize the Color Picker container

|.color-picker.e-dropdown-popup ul .e-item.e-palette-item|To customize the Color Picker pallete item

|.color-picker.e-dropdown-popup .e-container .e-switch|To customize the Color Picker switch control

|.color-picker.e-dropdown-popup .e-container .e-slider-preview|To customize the Color Picker slider control

How To

Hide control buttons in React Color picker component

ColorPicker can be rendered without control buttons (Apply/Cancel). In this case, while selecting a color, the

ColorPicker pop-up is closed and selected colors can be applied directly. To hide control buttons, set the [showButtons](#) property to `false`.

INDEX.JSX

```
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
//To hide control buttons.
function App() {
    return (<div id='container'>
        <div className='wrap'>
            <h4>Choose Color</h4>
            <ColorPickerComponent showButtons={false}/>
        </div>
    </div>);
}
;
export default App;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
//To hide control buttons.
function App() {
    return (
        <div id='container'>
            <div className='wrap'>
                <h4>Choose Color</h4>
                <ColorPickerComponent showButtons={false} />
            </div>
        </div>
    );
};
export default App;
ReactDOM.render(<App />, document.getElementById('element'));
```

Render palette alone in React Color picker component

To render the `Palette` alone in ColorPicker, specify the [mode](#) property as `Palette`, and set the [modeSwitcher](#) property to `false`.

In the following sample, the [showButtons](#) property is disabled to hide the control buttons and it renders only the `Palette` area.

INDEX.JSX

```
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
```



```
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  return (<div id='container'>
    <div className='wrap'>
      <h4>Choose Color</h4>
      <ColorPickerComponent mode="Palette" modeSwitcher={false}
showButtons={false}></ColorPickerComponent>
    </div>
  </div>);
}
;
export default App;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  return (
    <div id='container'>
      <div className='wrap'>
        <h4>Choose Color</h4>
        <ColorPickerComponent mode = "Palette" modeSwitcher={false}
showButtons ={false}></ColorPickerComponent>
      </div>
    </div>
  );
};
export default App;
ReactDOM.render(<App />, document.getElementById('element'));
```

> To render **Picker** alone specify the [mode](#) property as 'Picker'.

Colorpicker in dropdownbutton in React Color picker component

This section explains about how to render the ColorPicker in DropDownButton. The [target](#) property of the DropDownButton helps to achieve this scenario. To know about the usage of **target** property refer to [Popup templating](#) section.

In the below sample, the color picker is rendered as inline type by setting [inline](#) property as **true** and the rendered color picker wrapper is passed as a **target** to the DropDownButton to achieve the above scenario.

INDEX.JSX

```
{% raw %}
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import { DropDownButtonComponent } from '@syncfusion/ej2-react-splitbuttons';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  let ddb;
  let cp;
```

```

function onChange(args) {
    ddb.element.children[0].style.backgroundColor =
args.currentValue.rgba;
    closePopup();
}
function closePopup() {
    ddb.toggle();
}
function onCreated() {
    const parentElem = cp.element.parentElement;
    const cancelBtn = parentElem.querySelector('.e-cancel');
    cancelBtn.addEventListener('click', closePopup.bind(this));
}
function open() {
    var zIndex = (document.getElementsByClassName('e-color-picker-
tooltip')[0]).style.zIndex;
    var zIndexIntValue = parseInt(zIndex) + 2;
    (document.getElementsByClassName('e-color-picker-
tooltip')[0]).style.zIndex = zIndexIntValue.toString();
}
return (<div id='container'>
    <div className='wrap'>
        <h4>Choose Color</h4>
        <ColorPickerComponent ref={ (scope) => { cp = scope; }}
id='colorpicker' inline={true} change={onChange}/>
        <DropDownButtonComponent id='dropdownbtn' open={open} target='.e-
colorpicker-wrapper' ref={ (scope) => { ddb = scope; }} iconCss='e-
dropdownbtn-preview' created={onCreated}/>
    </div>
</div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import { ColorPickerComponent, ColorPickerEventArgs } from '@syncfusion/ej2-
react-inputs';
import { DropDownButtonComponent } from '@syncfusion/ej2-react-splitbuttons';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    let ddb: DropDownButtonComponent;
    let cp: ColorPickerComponent;
    function onChange(args: ColorPickerEventArgs): void {
        (ddb.element.children[0] as HTMLElement).style.backgroundColor =
args.currentValue.rgba;
        closePopup();
    }
    function closePopup(): void {
        ddb.toggle();
    }
    function onCreated() {
        const parentElem = cp.element.parentElement as HTMLElement;

```

```

    const cancelBtn = parentElem.querySelector('.e-cancel') as
    HTMLElement;
    cancelBtn.addEventListener('click', closePopup.bind(this));
  }
  function open() {
    var zIndex = (document.getElementsByClassName('e-color-picker-
    tooltip')[0] as HTMLElement).style.zIndex;
    var zIndexIntValue = parseInt(zIndex) + 2;
    (document.getElementsByClassName('e-color-picker-tooltip')[0] as
    HTMLElement).style.zIndex = zIndexIntValue.toString();
  }
  return (
    <div id='container'>
      <div className='wrap'>
        <h4>Choose Color</h4>
        <ColorPickerComponent ref={ (scope) => { cp = scope as
        ColorPickerComponent; }} id='colorpicker' inline={true} change={onChange}/>
        <DropDownButtonComponent id='dropdownbtn' open={open} target='.e-
        colorpicker-wrapper' ref={ (scope) => { ddb = scope as
        DropDownButtonComponent; }} iconCss='e-dropdownbtn-preview'
        created={onCreated}/>
      </div>
    </div>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));
{% enddraw %}

```

Customize colorpicker in React Color picker component

Custom palette

By default, the Palette will be rendered with default colors. To load custom colors in the palette, specify the colors in the [presetColors](#) property. To customize the color palette, add a custom class to palette tiles using [BeforeTileRender](#) event.

The following sample demonstrates the above functionalities.

INDEX.JSX

```

import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  let presets = {
    'custom1': ['#ef9a9a', '#e57373', '#ef5350',
      '#f44336', '#f48fb1', '#f06292',
      '#ec407a', '#e91e63', '#ce93d8',
      '#ba68c8', '#ab47bc', '#9c27b0',
      '#b39ddb', '#9575cd', '#7e57c2', '#673ab7'],
    'custom2': ['#9fa8da', '#7986cb', '#5c6bc0', '#3f51b5',
      '#90caf9', '#64b5f6', '#42a5f5', '#2196f3',
      '#81d4fa', '#4fc3f7', '#29b6f6', '#03a9f4',
      '#80deea', '#4dd0e1', '#26c6da', '#00bcd4'],
    'custom3': ['#80cbc4', '#4db6ac', '#26a69a', '#009688',
      '#a5d6a7', '#81c784', '#66bb6a', '#4caf50',
      '#c5e1a5', '#aed581', '#9ccc65', '#8bc34a', '#e6ee9c'],
  };

```

```

        '#DCE775', '#D4E157', '#CDDC39']
    };
    // Triggers before rendering each palette tile.
    function tileRender(args) {
        args.element.classList.add("e-icons");
        args.element.classList.add("e-custom-tile");
    }
    // riggers while selecting colors from palette.
    function change(args) {
        document.getElementById('preview').style.backgroundColor =
args.currentValue.hex;
    }
    return (<div id='container'>
        <div className='wrap'>
            <div id="preview"/>
            <h4>Select Color</h4>
            <ColorPickerComponent id='element' mode='Palette'
modeSwitcher={false} inline={true} showButtons={false} columns={4}
presetColors={presets} beforeTileRender={tileRender} change={change}/>
            </div>
        </div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { ColorPickerComponent, ColorPickerEventArgs, PaletteTileEventArgs }
from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    let presets: { [key: string]: string[] } = {
        'custom1': ['#ef9a9a', '#e57373', '#ef5350',
            '#f44336', '#f48fb1', '#f06292',
            '#ec407a', '#e91e63', '#ce93d8',
            '#ba68c8', '#ab47bc', '#9c27b0',
            '#b39ddb', '#9575cd', '#7e57c2', '#673ab7'],
        'custom2': ['#9fa8da', '#7986cb', '#5c6bc0', '#3f51b5',
            '#90caf9', '#64b5f6', '#42a5f5', '#2196f3',
            '#81d4fa', '#4fc3f7', '#29b6f6', '#03a9f4',
            '#80ddea', '#4dd0e1', '#26c6da', '#00bcd4'],
        'custom3': ['#80cbc4', '#4db6ac', '#26a69a', '#009688',
            '#a5d6a7', '#81c784', '#66bb6a', '#4caf50',
            '#c5e1a5', '#aed581', '#9ccc65', '#8bc34a',
            '#e6ee9c',
            '#dce775', '#d4e157', '#cddc39']
    };
    // Triggers before rendering each palette tile.
    function tileRender(args: PaletteTileEventArgs): void {
        args.element.classList.add("e-icons");
        args.element.classList.add("e-custom-tile");
    }
    // riggers while selecting colors from palette.
    function change(args: ColorPickerEventArgs): void {

```

```

        (document.getElementById('preview') as
HTMLInputElement).style.backgroundColor = args.currentValue.hex;
    }
    return (
        <div id='container'>
        <div className='wrap'>
            <div id="preview"/>
            <h4>Select Color</h4>
            <ColorPickerComponent id='element' mode='Palette'
modeSwitcher={false} inline={true} showButtons={false} columns={4}
presetColors={presets} beforeTileRender={tileRender} change = {change}/>
        </div>
        </div>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

Hide input area from picker

By default, the input area will be rendered in ColorPicker. To hide the input area from it, add `e-hide-value` class to ColorPicker using the [cssClass](#) property.

In the following sample, the ColorPicker is rendered without input area.

INDEX.JSX

```

import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
// To hide the input area
function App() {
    return (<div id='container'>
        <div className='wrap'>
            <h4>Choose Color</h4>
            <ColorPickerComponent cssClass="e-hide-value"
modeSwitcher={false}/>
        </div>
        </div>);
}
;
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
// To hide the input area
function App() {
    return (
        <div id='container'>
        <div className='wrap'>
            <h4>Choose Color</h4>
            <ColorPickerComponent cssClass="e-hide-value"
modeSwitcher={false} />

```

```

        </div>
      </div>
    );
  };
  export default App;
  ReactDOM.render(<App />, document.getElementById('element'));

```

Custom handle

Color picker handle shape and UI can be customized. Here, we have customized the handle as **svg icon**. The same way you can customize the handle based on your requirement.

The following sample show the customized color picker handle.

INDEX.JSX

```

import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  return (<div id='container'>
    <div className='wrap'>
      <h4>Choose Color</h4>
      <ColorPickerComponent id='colorpicker' value='#344aae'
cssClass='e-custom-picker' modeSwitcher={false}/>
    </div>
  </div>);
}
;
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  return (
    <div id='container'>
      <div className='wrap'>
        <h4>Choose Color</h4>
        <ColorPickerComponent id='colorpicker' value='#344aae'
cssClass='e-custom-picker' modeSwitcher={false}/>
      </div>
    </div>
  );
};
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

Custom primary button

By default, the applied color will be updated in primary button of the color picker. You can customize that as **icon**.

In the following sample, the `picker` icon is added to primary button and using [change](#) event the selected color will be updated in bottom portion of the icon.

INDEX.JSX

```
import { addClass } from '@syncfusion/ej2-base';
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    let previewIcon;
    let cp;
    let onCreated = onCreated.bind(this);
    let onChange = onChange.bind(this);
    function onChange(args) {
        previewIcon.style.borderBottomColor = args.currentValue.rgba;
    }
    function onCreated() {
        const elem = cp.element.nextElementSibling;
        previewIcon = elem.querySelector('.e-selected-color');
        addClass([previewIcon], 'e-icons');
    }
    return (<div id='container'>
        <div className='wrap'>
            <h4>Choose Color</h4>
            <ColorPickerComponent ref={(scope) => cp = scope}
id='colorpicker' created={onCreated} change={onChange}/>
        </div>
    </div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import { addClass } from '@syncfusion/ej2-base';
import { ColorPickerComponent, ColorPickerEventArgs } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    let previewIcon: HTMLElement;
    let cp: ColorPickerComponent;
    let onCreated: any = onCreated.bind(this);
    let onChange: any = onChange.bind(this);
    function onChange (args: ColorPickerEventArgs): void {
        previewIcon.style.borderBottomColor = args.currentValue.rgba;
    }
    function onCreated() {
        const elem = cp.element.nextElementSibling as HTMLElement;
        previewIcon = elem.querySelector('.e-selected-color') as HTMLElement;
        addClass([previewIcon], 'e-icons');
    }
    return (
        <div id='container'>
            <div className='wrap'>
```

```

        <h4>Choose Color</h4>
        <ColorPickerComponent ref= { (scope) => cp = scope as
ColorPickerComponent } id='colorpicker' created={onCreated}
change={onChange}/>
      </div>
    </div>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

> The Essential JS 2 provides a set of icons that can be loaded by applying **e-icons** class name to the element. You can also use third party icon to customize the primary button.

Display hex code in input

The color picker input element can be showcased in the place of primary button. The applied color hex code will be updated in the primary button input.

The following sample shows the color picker with input.

INDEX.JSX

```

import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  let colorPicker;
  let onCreated = onCreated.bind(this);
  function onCreated() {
    const cpElem = colorPicker.element.nextElementSibling;
    cpElem.insertBefore(colorPicker.element, cpElem.children[1]);
  }
  return (<div id='container'>
    <div className='wrap'>
      <h4>Choose Color</h4>
      <ColorPickerComponent ref={ (scope) => colorPicker = scope }
id='colorpicker' type='text' created={onCreated} className='e-input' />
    </div>
  </div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  let colorPicker: ColorPickerComponent;
  let onCreated: any = onCreated.bind(this);
  function onCreated(): void {
    const cpElem = colorPicker.element.nextElementSibling as HTMLElement;
    cpElem.insertBefore(colorPicker.element, cpElem.children[1]);
  }
  return (

```



```

        <div id='container'>
          <div className='wrap'>
            <h4>Choose Color</h4>
            <ColorPickerComponent ref={(scope) => colorPicker = scope as
ColorPickerComponent} id='colorpicker' type='text' created={onCreated}
className='e-input' />
          </div>
        </div>
      );
    }
  }
  export default App;
  ReactDOM.render(<App />, document.getElementById('element'));

```

Custom UI

The color picker UI can be customized in all possible ways. The following sample shows the excel like UI customization with help of SplitButton and Dialog component. In that by clicking the more colors option from color palette, the dialog contains color picker will open.

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ColorPickerComponent, PaletteTileEventArgs, ColorPickerEventArgs }
from '@syncfusion/ej2-react-inputs';
import { DialogComponent } from '@syncfusion/ej2-react-popups';
import { SplitButtonComponent, BeforeOpenCloseMenuEventArgs,
OpenCloseMenuEventArgs } from '@syncfusion/ej2-react-splitbuttons';
function App() {
  let splitIcon;
  let colorPicker;
  let pickerDlg;

  let animationSettings = { effect: 'Zoom' };

  function content(data) {
    function onPickerChange(args) {
      onPaletteChange(args);
      pickerDlg.hide();
    }

    return (
      <div className="dialogContent">
        <ColorPickerComponent id='picker' inline={true}
modeSwitcher={false} change={onPickerChange} ref={(scope) => { colorPicker =
scope; }}></ColorPickerComponent>
      </div>
    )
  }

  function onPaletteChange (args) {
    splitIcon = document.getElementById("split-btn").children[0];
    splitIcon.style.borderBottomColor = args.currentValue.rgba;
  }

  function onDdPopupOpen(args) {
    args.element.children[1].addEventListener('click', openPickerDlg);
  }

```

```

    }

    function onBeforeDdPopupClose (args) {
        args.element.children[1].removeEventListener('click', openPickerDlg);
    }

    function openPickerDlg() {
        pickerDlg.show();
    }

    function pickerDlgOpen() {
        colorPicker.refresh();
        colorPicker.element.nextElementSibling.querySelector('.e-ctrl-btn .e-cancel').addEventListener('click', pickerDlgClose);
    }

    function pickerDlgClose() {
        pickerDlg.hide();
    }

    function onSplitBtnCreated() {
        splitIcon = document.getElementById("split-btn").children[0];
    }

    return (
        <div id='container'>
            <div className='wrap'>
                <ul id="target">
                    <li className="e-item e-palette-item">
                        <ColorPickerComponent id='palette' mode='Palette'
inline={true} showButtons={false} modeSwitcher={false}
change={onPaletteChange}></ColorPickerComponent>
                    </li>
                    <li className="e-item">
                        <span className="e-menu-icon"></span>
                        More colors...
                    </li>
                </ul>
                <h4>Select color</h4>
                <SplitButtonComponent id='split-btn' created={onSplitBtnCreated}
iconCss='e-icons e-font-icon' target='#target' open={onDdPopupOpen}
beforeClose={onBeforeDdPopupClose}></SplitButtonComponent>
                <DialogComponent id='picker-dialog' cssClass='e-dlg-picker'
isModal={true} height='336px' width='270px' ref={ (dialog) => { pickerDlg =
dialog; }} target='.wrap' content= {content} overlayClick={pickerDlgClose}
open={pickerDlgOpen} visible={false}
animationSettings={animationSettings}></DialogComponent>
            </div>
        </div>
    );
}

export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ColorPickerComponent, PaletteTileEventArgs, ColorPickerEventArgs }
from '@syncfusion/ej2-react-inputs';
import { DialogComponent } from '@syncfusion/ej2-react-popups';
import { SplitButtonComponent, BeforeOpenCloseMenuEventArgs,
OpenCloseMenuEventArgs } from '@syncfusion/ej2-react-splitbuttons';
function App() {
    let splitIcon: HTMLElement;
    let colorPicker: ColorPickerComponent;
    let pickerDlg: DialogComponent;

    let animationSettings: Object = { effect: 'Zoom' };

    function content(data: any): JSX.Element {
        function onPickerChange(args: ColorPickerEventArgs): void {
            onPaletteChange(args);
            pickerDlg.hide();
        }

        return (
            <div className="dialogContent">
                <ColorPickerComponent id='picker' inline={true}
modeSwitcher={false} change={onPickerChange} ref={(scope) => { colorPicker =
scope as ColorPickerComponent; }}></ColorPickerComponent>
                </div>
            )
        )

        function onPaletteChange (args: ColorPickerEventArgs): void {
            splitIcon = document.getElementById("split-btn")!.children[0] as
HTMLElement;
            splitIcon.style.borderBottomColor = args.currentValue.rgba;
        }

        function onDdPopupOpen(args: OpenCloseMenuEventArgs): void {
            args.element.children[1].addEventListener('click', openPickerDlg);
        }

        function onBeforeDdPopupClose (args: BeforeOpenCloseMenuEventArgs): void
{
            args.element.children[1].removeEventListener('click', openPickerDlg);
        }

        function openPickerDlg(): void {
            pickerDlg.show();
        }

        function pickerDlgOpen(): void {
            colorPicker.refresh();
            colorPicker.element.nextElementSibling!.querySelector('.e-ctrl-btn
.e-cancel')!.addEventListener('click', pickerDlgClose);
        }

        function pickerDlgClose(): void {
            pickerDlg.hide();
        }
    }
}

```

```

function onSplitBtnCreated(): void {
    splitIcon = document.getElementById("split-btn")!.children[0] as
    HTMLInputElement;
}

return (
    <div id='container'>
        <div className='wrap'>
            <ul id="target">
                <li className="e-item e-palette-item">
                    <ColorPickerComponent id='palette' mode='Palette'
inline={true} showButtons={false} modeSwitcher={false}
change={onPaletteChange}></ColorPickerComponent>
                </li>
                <li className="e-item">
                    <span className="e-menu-icon"></span>
                    More colors...
                </li>
            </ul>
            <h4>Select color</h4>
            <SplitButtonComponent id='split-btn' created={onSplitBtnCreated}
iconCss='e-icons e-font-icon' target='#target' open={onDdPopupOpen}
beforeClose={onBeforeDdPopupClose}></SplitButtonComponent>
            <DialogComponent id='picker-dialog' cssClass='e-dlg-picker'
isModal={true} height='336px' width='270px' ref={ (dialog) => { pickerDlg =
dialog as DialogComponent; }} target='.wrap' content= {content}
overlayClick={pickerDlgClose} open={pickerDlgOpen} visible={false}
animationSettings={animationSettings}></DialogComponent>
        </div>
    </div>
);
}

export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

STYLES.CSS

```

#loader {
    color: #008cff;
    font-family: 'Helvetica Neue', 'calibiri';
    font-size: 14px;
    height: 40px;
    left: 45%;
    position: absolute;
    top: 45%;
    width: 30%;
}

.wrap {
    margin: 0 auto;
    width: 100%;
    height: 100%;
    min-height: 350px;
    text-align: center;
}

```

```

/* Primary button icon preview */
.e-btn-icon.e-font-icon {
  border-bottom-style: solid;
  border-bottom-width: 3px;
}
/* Primary button icon */
.e-btn-icon.e-font-icon::before {
  content: '\e34c';
}
.e-colorpicker-wrapper.e-hide-palette {
  display: none;
}
.e-dropdown-popup ul .e-item:first-child.e-palette-item {
  height: auto;
  padding: 0;
}
.e-dlg-picker.e-dialog .e-dlg-content {
  padding: 0;
  background-color: transparent;
}
/* Sets ColorPicker height */
.e-dlg-picker.e-dialog {
  max-height: 336px !important;
  background-color: transparent;
}
/* More colors li icon customization */
.e-dropdown-popup ul .e-item:last-child .e-menu-icon {
  height: 24px;
  margin-top: 6px;
  width: 24px;
  background-image: linear-gradient(to bottom, #fff 0, #000 100%);
  background-color: #0450c2;
  background-blend-mode: hard-light;
}
h4 {
  font-family: 'Helvetica Neue', 'Helvetica', 'Arial', 'sans-serif';
  font-size: 14px;
}

```

Handle no color support in React Color picker component

The ColorPicker component supports no color functionality. By clicking the no color tile from palette, the selected color becomes **empty** and considered as no color has been selected from color picker.

Default no color

To achieve this, set **noColor** property as **true**.

In the following sample, the first tile of the color palette represents the no color tile. By clicking the no color tile you can achieve the above functionalities.

INDEX.JSX

```

import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
import { useEffect } from "react";
function App() {

```

```

let preview;
function onChange(args) {
    preview.style.backgroundColor = args.currentValue.hex;
    preview.textContent = args.currentValue.hex ? args.currentValue.hex :
    'No color';
}
useEffect(() => {
    preview = document.getElementById('preview');
    preview.style.backgroundColor = '#ba68c8';
    preview.textContent = '#ba68c8';
}, []);
return (<div id='container'>
    <div className='wrap'>
        <div id='preview' />
        <h4>Select Color</h4>
        <ColorPickerComponent id='colorpicker' value='#ba68c8'
mode='Palette' noColor={true} showButtons={false} modeSwitcher={false}
change={onChange} />
    </div>
    </div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { ColorPickerComponent, ColorPickerEventArgs } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
import { useEffect } from "react";
function App() {
    let preview: HTMLElement;
    function onChange (args: ColorPickerEventArgs): void {
        preview.style.backgroundColor = args.currentValue.hex;
        preview.textContent = args.currentValue.hex ? args.currentValue.hex :
    'No color';
    }
    useEffect(() => {
        preview = document.getElementById('preview') as HTMLElement;
        preview.style.backgroundColor = '#ba68c8';
        preview.textContent = '#ba68c8';
    }, []);
    return (
        <div id='container'>
            <div className='wrap'>
                <div id='preview' />
                <h4>Select Color</h4>
                <ColorPickerComponent id='colorpicker' value='#ba68c8'
mode='Palette' noColor={true} showButtons={false} modeSwitcher={false}
change={onChange} />
            </div>
            </div>
        );
    }
export default App;

```

```
ReactDOM.render(<App />, document.getElementById('element'));
```

If the [noColor](#) property is enabled, make sure to disable the [modeswitcher](#) property.

Custom no color

The following sample show the color palette with custom no color option.

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
import { ColorPickerComponent, PaletteTileEventArgs, ColorPickerEventArgs }
from '@syncfusion/ej2-react-inputs';
import { SplitButtonComponent } from '@syncfusion/ej2-react-splitbuttons';
function App() {
    let preview;
    let splitBtn;
    let colorPicker;
    let presets = {
        'custom': ['#f44336', '#e91e63', '#9c27b0', '#673ab7', '#2196f3',
        '#03a9f4', '#00bcd4', '#009688', '#8bc34a', '#cddc39', '#ffeb3b', '#ffc107']
    };
    function beforeTileRender(args) {
        args.element.classList.add('e-custom-tile');
    }
    function onChange (args) {
        preview = document.getElementById('preview');
        document.querySelector(".e-split-btn .e-picker-
icon").style.borderBottomColor = args.currentValue.hex;
        preview.style.backgroundColor = args.currentValue.hex;
        preview.textContent = args.currentValue.hex;
        if (splitBtn.element.getAttribute("aria-expanded")) {
            splitBtn.toggle();
            splitBtn.element.focus();
        }
    }
    function onCreated() {
        preview = document.getElementById('preview');
        preview.style.backgroundColor = '#ba68c8';
        preview.textContent = '#ba68c8';
        document.getElementById('no-color').onclick = () => {
            //sets color picker value property to null
            colorPicker.setProperties({ 'value': '' }, true);
            document.querySelector('.e-split-btn .e-picker-
icon').style.borderBottomColor = 'transparent';
            preview.textContent = 'No color';
            preview.style.backgroundColor = 'transparent';
        }
    }
    return (
        <div id='container'>
            <div className='wrap'>
                <ul id="target">
                    <li className="e-item e-palette-item">
                        <ColorPickerComponent id='colorpicker' ref={(scope) => {
colorPicker = scope; }} value='#f44336' mode='Palette' inline={true}>
```

```

columns={4} presetColors={presets} showButtons={false} modeSwitcher={false}
beforeTileRender={beforeTileRender} change={onChange}
created={onCreated}></ColorPickerComponent>
    </li>
    <li className="e-item" id="no-color">
        <span className="e-menu-icon e-nocolor"></span>
        No color
    </li>
</ul>
<div>
    <div id='preview'></div>
    <h4>Select color</h4>
    <SplitButtonComponent id='splitbtn' iconCss='e-cp-icons e-
picker-icon' target='#target' ref={(scope) => { splitBtn = scope;
}}></SplitButtonComponent>
    </div>
</div>
</div>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
import { ColorPickerComponent, PaletteTileEventArgs, ColorPickerEventArgs }
from '@syncfusion/ej2-react-inputs';
import { SplitButtonComponent } from '@syncfusion/ej2-react-splitbuttons';
function App() {
    let preview: HTMLElement;
    let splitBtn: SplitButtonComponent;
    let colorPicker: ColorPickerComponent;
    let presets: { [key: string]: string[] } = {
        'custom': ['#f44336', '#e91e63', '#9c27b0', '#673ab7', '#2196f3',
        '#03a9f4', '#00bcd4', '#009688', '#8bc34a', '#cddc39', '#ffeb3b', '#ffc107']
    };
    function beforeTileRender(args: PaletteTileEventArgs): void {
        args.element.classList.add('e-custom-tile');
    }
    function onChange (args: ColorPickerEventArgs): void {
        preview = document.getElementById('preview') as HTMLElement;
        (document.querySelector(".e-split-btn .e-picker-icon") as
        HTMLElement).style.borderBottomColor = args.currentValue.hex;
        preview.style.backgroundColor = args.currentValue.hex;
        preview.textContent = args.currentValue.hex;
        if (splitBtn.element.getAttribute("aria-expanded")) {
            splitBtn.toggle();
            splitBtn.element.focus();
        }
    }
    function onCreated(): void {
        preview = document.getElementById('preview') as HTMLElement;
        preview.style.backgroundColor = '#ba68c8';
        preview.textContent = '#ba68c8';
    }
}

```



```

        document.getElementById('no-color')!.onclick = (): void => {
            //sets color picker value property to null
            colorPicker.setProperties({ 'value': '' }, true);
            (document.querySelector('.e-split-btn .e-picker-icon') as
HTMLInputElement).style.borderBottomColor = 'transparent';
            preview.textContent = 'No color';
            preview.style.backgroundColor = 'transparent';
        }
    }
    return (
        <div id='container'>
            <div className='wrap'>
                <ul id="target">
                    <li className="e-item e-palette-item">
                        <ColorPickerComponent id='colorpicker' ref={(scope) => {
colorPicker = scope as ColorPickerComponent; }} value='#f44336'
mode='Palette' inline={true} columns={4} presetColors={presets}
showButtons={false} modeSwitcher={false} beforeTileRender={beforeTileRender}
change={onChange} created={onCreated}></ColorPickerComponent>
                    </li>
                    <li className="e-item" id="no-color">
                        <span className="e-menu-icon e-nocolor"></span>
                        No color
                    </li>
                </ul>
                <div>
                    <div id='preview'></div>
                    <h4>Select color</h4>
                    <SplitButtonComponent id='splitbtn' iconCss='e-cp-icons e-
picker-icon' target='#target' ref={(scope) => { splitBtn = scope as
SplitButtonComponent; }}></SplitButtonComponent>
                </div>
            </div>
        </div>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

STYLES.CSS

```

#loader {
    color: #008cff;
    font-family: 'Helvetica Neue', 'calibiri';
    font-size: 14px;
    height: 40px;
    left: 45%;
    position: absolute;
    top: 45%;
    width: 30%;
}
@font-face {
    font-family: 'paint';
    src:
url(data:application/x-font-ttf;charset=utf-
8;base64,AAEAAAIAIAAAwAgTlMvMj0gSRIAAAEoAAAAVmNtYXNlEODVAAABiAAAADZnbHlIZD+

```

```

uwAAAcgAAADMaGVhZBKhhHQAADQAAAAANmhoZWEHjANrAAAArAAAACRobXR4B+j/8wAAAYAAAAIb
G9jYQBMAAAAAAHAAAAABmlheHABDgBKAAABCAAAACBuYw1ln6hzswAAApQAAAIncG9zdEkLMmUAAA
SkAAAAngABAAADUv9qAFoEAP/z//4D6gABAAAAAAAAAAAAAAAAAAAAgABAAAAAQAAZfc6F8PPPU
ACwPoAAAAANfSn9kAAAAA19Kf2f/z//wD6gPhAAAAACAACAAAAAAAAAAAAEAAAACAD4AAgAAAAAAAgAA
AAoACgAAAP8AAAAAAAAAAQP0AZAABQAAAnoCvAAAAIwCegK8AAAB4AAxAQIAAAIABQMAAAAAAAAA
AAAAAAAAAAAAAAAAAUGZFZABA5wDnAANS/2oAWgPhAJYAAAABAAAAAAAAABAAAAAPo//MAAAACAA
AAAwAAABQAAwABAAAAFAAEACIAAAAEAAQAAQAA5wD//wAA5wD//wAAAAEABAAAAAEAAAAAAAAAZgA
AAAL/8//8A+oD4QAKAD0AAAEWBgceATc1JiQHJTMmNjceARcVJx4BBx4BFQ4BIiYnNDY3PgEvAS4B
Iw4BBwEGHgI3AT4BLwE1LgEnDgEDEiRlCgulCxP+8RT+GyYDQFxoZQwTBQEDDxEBJzonAREOCQkPJ
Q4cDBcdAf6oG1a3nx8BWQ4RHKADeG1oWwHTLHVwYVml6Kx1BHEqfwYFqWUHEX4tDAocEx0nJx0RHg
oVUDQpDgsBFAH+px2guFUaAVkNOiCgCXnhCAWOAAAAAAAAAEgDeAAEAAAAAAAAAAQAAAAEAAAAAAE
ABQABAAEAAAAAAIABwAGAAEAAAAAAAMABQANAAEAAAAAAQAQASAAEAAAAAAAUACwAXAAEAAAAA
AAYABQAIAAEAAAAAAAOALAAAnAAEAAAAAAASAEgBTAAMAAQQAIAAAAgBLAAMAAQQAIAEACgBnAAMAA
QQJAAIADgBxAAMAAQQAIAAMACgB/AAMAAQQAIAAQACgCJAAMAAQQAIAAUAFgCTAAMAAQQAIAAYACgCpAA
MAAQQAIAAoAWACzAAMAAQQAIAAsAJAELIHBhaW50UmVndWxhcncBhaW50cGFpbmRWZXJzaW9uIDEuMHB
haW50Rm9udCBnZW51cmF0ZWQgdXNpbmcgU3luY2Zlc2lvbiBNZXRYbyBTdHVkaW93d3cuc3luY2Zl
c2lvbi5jb20AIABwAGEAaQBuAHQAUGBlAGcAdQBsAGEAcgBwAGEAaQBuAHQAACABhAGkAbgB0AFYAZ
QByAHMAaQBVAG4AIAAxAC4AMABwAGEAaQBuAHQARgBvAG4AdAAgAGcAZQBwAGUAcgBhAHQAZQBkAC
AAQzBzAGkAbgBnACAAUwB5AG4AYwBmAHUAcwBpAG8AbgAgAE0AZQB0AHIAbwAgAFMAdAB1AGQAaQB
vAHcAdwB3AC4AcwB5AG4AYwBmAHUAcwBpAG8AbgAuAGMabwBtAAAAAAIAAAAAAAAAACgAAAAAAAA
AAAAAAAAAAAAAAAAAAAgECAQMADHBhaW50LWJlY2tldAAAAA=) format('trueType');
font-weight: normal;
font-style: normal;
}
.e-cp-icons {
  font-family: 'paint' !important;
  speak: none;
  font-size: 55px;
  font-style: normal;
  font-weight: normal;
  font-variant: normal;
  text-transform: none;
  line-height: 1;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
}
/* Preview area styles */
#preview {
  border: 1px solid;
  height: 40px;
  line-height: 40px;
  width: 100%;
}
.wrap {
  margin: 0 auto;
  width: 300px;
  text-align: center;
}
/* ColorPicker customization */
.e-dropdown-popup ul#target {
  padding: 0;
}
.e-dropdown-popup ul .e-item.e-palette-item {
  height: auto;
  padding: 0;
}
.e-btn-icon.e-picker-icon {

```

[illegible]

Disabled in React Color picker component

To achieve disabled state in ColorPicker, set the `disabled` property to `true`. The ColorPicker pop-up cannot be accessed in disabled state.

The following example shows the disabled state of ColorPicker component.

INDEX.JSX

```
import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  return (<div id='container'>
    <div className='wrap'>
```

```

        <h4>Choose Color</h4>
        <ColorPickerComponent disabled={true}/>
      </div>
    </div>);
  }
;
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { ColorPickerComponent } from '@syncfusion/ej2-react-inputs';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  return (
    <div id='container'>
      <div className='wrap'>
        <h4>Choose Color</h4>
        <ColorPickerComponent disabled={true}/>
      </div>
    </div>
  );
};
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

Ej1 api migration in React Color picker component

This article describes the API migration process of ColorPicker component from Essential JS 1 to Essential JS 2.

Properties

{% raw %}

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| Default | **property:** *value*

 <EJ.ColorPicker id="colorpicker" value="#278787"></EJ.ColorPicker> | **property:** *value*

 <ColorPickerComponent id="colorpicker" value="#278787"></ColorPickerComponent> |

| Inline mode color picker | **property:** *displayInline*

 <EJ.ColorPicker id="colorpicker" displayInline={true}></EJ.ColorPicker> | **property:** *inline*

 <ColorPickerComponent id="colorpicker" inline={true}></ColorPickerComponent> |

| Adding custom class | **property:** *cssClass*

 <EJ.ColorPicker id="colorpicker" cssClass="custom-class"></EJ.ColorPicker> | **property:** *cssClass*

 <ColorPickerComponent id="colorpicker" cssClass="custom-class"></ColorPickerComponent> |

| Disable the ColorPicker component | **property:** *enabled*

 <EJ.ColorPicker id="colorpicker" enabled={false}></EJ.ColorPicker> | **property:** *disabled*

 <ColorPickerComponent id="colorpicker" disabled={true}></ColorPickerComponent> |

| To display custom text in button elements | **property:** *buttonText*

 var buttonText = { apply: "Apply", cancel: "Cancel", swatches: "Swatches" };
 <EJ.ColorPicker id="colorpicker" buttonText={buttonText}></EJ.ColorPicker> | Not Applicable |

| To display customized text or content when mouse over the color picker elements | **property:** *tooltipText*

 var tooltipText = { switcher: "Switch", currentColor: "New Color", selectedColor: "Old Color" };
 <EJ.ColorPicker id="colorpicker" tooltipText={tooltipText}></EJ.ColorPicker> | Not Applicable |

| Disable / hide opacity | **property:** *enableOpacity*

 <EJ.ColorPicker id="colorpicker" enableOpacity={false}></EJ.ColorPicker> | **property:** *enableOpacity*

 <ColorPickerComponent id="colorpicker" enableOpacity={false}></ColorPickerComponent> |

| ColorPicker Button mode | **property:** *buttonMode*

 <EJ.ColorPicker id="colorpicker" buttonMode="Dropdown"></EJ.ColorPicker> | Not Applicable |

| To show / hide the control (apply / cancel) buttons | **property:** *showApplyCancel*

 <EJ.ColorPicker id="colorpicker" showApplyCancel={false}></EJ.ColorPicker> | **property:** *showButtons*

 <ColorPickerComponent id="colorpicker" showButtons={false}></ColorPickerComponent> |

| To show / hide the clear button | **property:** *showClearButton*

 <EJ.ColorPicker id="colorpicker" showClearButton={false}></EJ.ColorPicker> | Not Applicable |

| Show / hide the mode (picker / palette) switcher | **property:** *showSwitcher*

 <EJ.ColorPicker id="colorpicker" showSwitcher={false}></EJ.ColorPicker> | **property:** *modeSwitcher*

 <ColorPickerComponent id="colorpicker" modeSwitcher={false}></ColorPickerComponent> |

| To show / hide the preview area | **property:** *showPreview*

 <EJ.ColorPicker id="colorpicker" showPreview={false}></EJ.ColorPicker> | Not Applicable |

| To show / hide the recent selected color list | **property:** *showRecentColors*

 <EJ.ColorPicker id="colorpicker" showRecentColors={true}></EJ.ColorPicker> | Not Applicable |

| To show / hide the color picker slider tooltip | **property:** *showTooltip*

 <EJ.ColorPicker id="colorpicker" showTooltip={false}></EJ.ColorPicker> | Not Applicable |

| Custom icon in dropdown control color area | **property:** *toolIcon*

 <EJ.ColorPicker id="colorpicker" toolIcon="e-font-icon"></EJ.ColorPicker> | Not Applicable |

| ColorPicker mode | **property:** *modelType*

 <EJ.ColorPicker id="colorpicker" modelType="Picker"></EJ.ColorPicker> | **property:** *mode*

 <ColorPickerComponent id="colorpicker" mode="Palette"></ColorPickerComponent> |

| Opacity value | **property:** *opacityValue*

 <EJ.ColorPicker id="colorpicker" opacityValue={80}></EJ.ColorPicker> | Not Applicable |

| Number of columns in color palette | **property:** *columns*

 <EJ.ColorPicker id="colorpicker" columns={10}></EJ.ColorPicker> | **property:** *columns*

 <ColorPickerComponent id="colorpicker" columns={15}></ColorPickerComponent> |

Custom colors | **property:** *palette*

 var colors = { "ffffff", "ffccff", "ff99ff", "ff66ff", "ff33ff", "ff00ff", "ccffff", "ccccff", "cc99ff", "cc66ff", "cc33ff", "cc00ff", "99ffff", "99ccff", "9999ff", "9966ff", "9933ff", "9900ff", "ffffcc", "ffcccc" };
 <EJ.ColorPicker id="colorpicker" palette="CustomPalette" modelType="Palette" custom={colors}></EJ.ColorPicker> | **property:** *presetColors*

 constructor(props: {}) {
 this.colors = {
 'custom': ["ffffff", "ffccff", "ff99ff", "ff66ff", "ff33ff", "ff00ff", "ccffff", "ccccff", "cc99ff", "cc66ff", "cc33ff", "cc00ff", "99ffff", "99ccff", "9999ff", "9966ff", "9933ff", "9900ff", "ffffcc", "ffcccc"];
 }
 <ColorPickerComponent id="colorpicker" presetColors={this.colors}></ColorPickerComponent> |

| Rendering palette from the predefined set of palettes | **property:** *presetType*

 <EJ.ColorPicker id="colorpicker" modelType="Palette" presetType="FlatColors"></EJ.ColorPicker> | Not Applicable |

| No color option in color palette | Not Applicable | **property:** *noColor*

 <ColorPickerComponent id="colorpicker" noColor={true} modeSwitcher={false} mode="Palette"></ColorPickerComponent> |

| Localization | **property:** *locale*

 ej.ColorPicker.Locale["zh-CN"] = {
 buttonText: {
 apply: "应用",
 cancel: "取消",
 swatches: "色板"
 },
 tooltipText: {
 switcher: "切换器",
 addButton: "添加颜色",
 basic: "基本"
 }
 }
 <EJ.ColorPicker id="colorpicker" locale="zh-CN"></EJ.ColorPicker> | **property:** *locale*

 L10n.load({
 'ar': {
 "colorpicker": {
 "Apply": "تطبيق",
 "Cancel": "إلغاء",
 "ModeSwitcher": "مفتاح كهربائي الوضع"
 }
 }
 });
 <ColorPickerComponent id="colorpicker" locale="ar"></ColorPickerComponent> |

| Right to left | **property:** *enableRTL*

 <EJ.ColorPicker id="colorpicker" enableRTL={true}></EJ.ColorPicker> | **property:** *enableRtl*

 <ColorPickerComponent id="colorpicker" enableRtl={true}></ColorPickerComponent> |

Methods

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

— — — — —

```
| Method to open color picker popup | Method: show <br/><br/> <EJ.ColorPicker
id="colorpicker"></EJ.ColorPicker> <br/> var colorPickerObj =
$('#colorpicker').ejColorPicker('instance'); <br/> colorPickerObj.show(); | Method: toggle <br/><br/>
<ColorPickerComponent id="colorpicker" ref = {(scope) => {this.colorPickerObj =
scope}}></ColorPickerComponent> <br/> constructor(props: {}) { <br/> &#160;
this.colorPickerObj.toggle(); <br/> } |
```

```
| Method to close color picker popup | Method: hide <br/><br/> <EJ.ColorPicker  
id="colorpicker"></EJ.ColorPicker> <br/> var colorPickerObj =  
$('#colorpicker').ejColorPicker('instance'); <br/> colorPickerObj.hide(); | Method: toggle <br/><br/>  
<ColorPickerComponent id="colorpicker" ref = {(scope) => {this.colorPickerObj =
```

```
scope}}></ColorPickerComponent> <br/> constructor(props: {}) { <br/> &#160;
this.colorPickerObj.toggle(); <br/> } |
```

| Enable the color picker control | **Method:** *enable*

 <EJ.ColorPicker
id="colorpicker"></EJ.ColorPicker>
 var colorPickerObj =
\$('#colorpicker').ejColorPicker('instance');
 colorPickerObj.enable(); | Not Applicable |

| Disables the color picker control | **Method:** *disable*

 <EJ.ColorPicker
id="colorpicker"></EJ.ColorPicker>
 var colorPickerObj =
\$('#colorpicker').ejColorPicker('instance');
 colorPickerObj.disable(); | Not Applicable |

| Method returns the selected color value as hex code | **Method:** *getValue*

 <EJ.ColorPicker
id="colorpicker"></EJ.ColorPicker>
 var colorPickerObj =
\$('#colorpicker').ejColorPicker('instance');
 colorPickerObj.getValue(); | **Method:** *getValue*

 <ColorPickerComponent id="colorpicker" ref = {(scope) => {this.colorPickerObj =
scope}}></ColorPickerComponent>
 constructor(props: {}) {

this.colorPickerObj.getValue();
 } |

| Method returns the selected color value in RGB format | **Method:** *getColor*

<EJ.ColorPicker id="colorpicker"></EJ.ColorPicker>
 var colorPickerObj =
\$('#colorpicker').ejColorPicker('instance');
 colorPickerObj.getColor(); | **Method:** *getValue*

 <ColorPickerComponent id="colorpicker" ref = {(scope) => {this.colorPickerObj =
scope}}></ColorPickerComponent>
 constructor(props: {}) {

this.colorPickerObj.getValue(null, 'RGB');
 } |

| Method convert the color value from hexCode to RGB | **Method:** *hexCodeToRGB*

<EJ.ColorPicker id="colorpicker"></EJ.ColorPicker>
 var colorPickerObj =
\$('#colorpicker').ejColorPicker('instance');
 colorPickerObj.hexCodeToRGB("#278787"); | **Method:**
getValue

 <ColorPickerComponent id="colorpicker" ref = {(scope) =>
{this.colorPickerObj = scope}}></ColorPickerComponent>
 constructor(props: {}) {

 this.colorPickerObj.getValue("#278787", 'RGB');
 } |

| Method convert the color value from RGB to Hex code | **Method:** *RGBToHEX*

<EJ.ColorPicker id="colorpicker"></EJ.ColorPicker>
 var colorPickerObj =
\$('#colorpicker').ejColorPicker('instance');
 colorPickerObj.RGBToHEX({r:38,g:133,b:133}); | **Method:**
getValue

 <ColorPickerComponent id="colorpicker" ref = {(scope) =>
{this.colorPickerObj = scope}}></ColorPickerComponent>
 constructor(props: {}) {

 this.colorPickerObj.getValue("rgb(38,133,133)", 'Hex');
 } |

| Method convert the color value from RGB to HSV | **Method:** *RGBToHSV*

 <EJ.ColorPicker
id="colorpicker"></EJ.ColorPicker>
 var colorPickerObj =
\$('#colorpicker').ejColorPicker('instance');
 colorPickerObj.RGBToHSV({h:230,s:98,v:98}); | **Method:**
getValue

 <ColorPickerComponent id="colorpicker" ref = {(scope) =>
{this.colorPickerObj = scope}}></ColorPickerComponent>
 constructor(props: {}) {

 this.colorPickerObj.getValue("rgb(180,71.1,52.9)", 'HSV');
 } |

| Method convert the color value from HSV to RGB | **Method:** *HSVToRGB*

 <EJ.ColorPicker
id="colorpicker"></EJ.ColorPicker>
 var colorPickerObj =
\$('#colorpicker').ejColorPicker('instance');
 colorPickerObj.HSVToRGB({h:230,s:98,v:98}); | **Method:**
getValue

 <ColorPickerComponent id="colorpicker" ref = {(scope) =>

```
{this.colorPickerObj = scope}}></ColorPickerComponent> <br/> constructor(props: {}) { <br/>
&#160; this.colorPickerObj.getValue("hsv(180,71.1,52.9)", 'RGB'); <br/> } |
```

Events

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| Event triggers before opening the ColorPicker popup | Not Applicable | **Event:** *beforeOpen*

<ColorPickerComponent id="colorpicker"

```
beforeOpen={this.beforeOpen.bind(this)}></ColorPickerComponent> <br/> beforeOpen(args) {
<br/> &#160; &#160; / code block */ <br/> } |
```

| Event triggers before closing the ColorPicker popup | Not Applicable | **Event:** *beforeClose*

<ColorPickerComponent id="colorpicker"

```
beforeClose={this.beforeClose.bind(this)}></ColorPickerComponent> <br/> beforeClose(args) {
<br/> &#160; &#160; / code block */ <br/> } |
```

| Event triggers after opening the ColorPicker popup | **Event:** *open*

 <EJ.ColorPicker
id="colorpicker" open={open}></EJ.ColorPicker>
 function open(args) {
 /
code block /
 } | **Event:** *open*

 <ColorPickerComponent id="colorpicker"
open={this.open.bind(this)}></ColorPickerComponent>
 open(args) {
 / code
block /
 } |

| Event triggers after closing the ColorPicker popup | **Event:** *close*

 <EJ.ColorPicker
id="colorpicker" close={close}></EJ.ColorPicker>
 function close(args) {
 /
code block */
 } | Not Applicable |

| Event triggers once the component rendering is completed | **Event:** *create*

<EJ.ColorPicker id="colorpicker" create={create}></EJ.ColorPicker>
 function create(args) {

 / code block /
 } | **Event:** *created*

 <ColorPickerComponent
id="colorpicker" created={this.created.bind(this)}></ColorPickerComponent>
 created() {

 / code block /
 } |

| Event triggers once the color picker control is destroyed | **Event:** *destroy*

 <EJ.ColorPicker
id="colorpicker" destroy={destroy}></EJ.ColorPicker>
 function destroy(args) {

 / code block */
 } | Not Applicable |

| Event triggers before Switching between Picker / Palette mode | Not Applicable | **Event:**
beforeModeSwitch

 <ColorPickerComponent id="colorpicker"
beforeModeSwitch={this.beforeModeSwitch.bind(this)}></ColorPickerComponent>

beforeModeSwitch(args) {
 / code block */
 } |

| Event triggers after color value has been selected | **Event:** *select*

 <EJ.ColorPicker
id="colorpicker" select={select}></EJ.ColorPicker>
 function select(args) {

 / code block /
 } | **Event:** *select*

 <ColorPickerComponent id="colorpicker"
select={this.select.bind(this)}></ColorPickerComponent>
 select(args) {
 /
code block /
 } |

| Event triggers after color value has been changed | **Event:** *change*

 <EJ.ColorPicker
id="colorpicker" change={change}></EJ.ColorPicker>
 function change(args) {

 / code block /
 } | **Event:** *change*

 <ColorPickerComponent id="colorpicker"


```
change={this.change.bind(this)}></ColorPickerComponent> <br/> change(args) { <br/> &#160; &#160;  
/ code block / <br/> } |
```

```
{% endraw %}
```

ComboBox

Getting Started

This section explains how to create a simple [Link to the Video](#) component and configure its available functionalities in React.

To get start quickly with React ComboBox, you can check on this video:

Dependencies

The following list of dependencies are required to use the **ComboBox** component in your application.

```
`javascript  
|-- @syncfusion/ej2-react-dropdowns  
|-- @syncfusion/ej2-base  
|-- @syncfusion/ej2-data  
|-- @syncfusion/ej2-react-base  
|-- @syncfusion/ej2-dropdowns  
|-- @syncfusion/ej2-lists  
|-- @syncfusion/ej2-inputs  
|-- @syncfusion/ej2-navigations  
|-- @syncfusion/ej2-popups  
|-- @syncfusion/ej2-buttons  
`,`
```

Installation and configuration

You can use [Create-react-app](#) to setup the applications. To install **create-react-app** run the following command.

```
`bash  
npm install -g create-react-app  
`,`
```

Start a new project using create-react-app command as follows

```
`bash  
create-react-app quickstart --scripts-version=react-scripts-ts  
cd quickstart  
`,`
```

Adding syncfusion packages

All the available Essential JS 2 packages are published in [npmjs.com](https://www.npmjs.com) public registry. You can choose the component that you want to install.

To install ComboBox component, use the following command

```
`bash
npm install @syncfusion/ej2-react-dropdowns --save
`
```

Adding ComboBox component

Now, you can start adding ComboBox component in the application. For getting started, add the ComboBox component in `src/App.tsx` file using following code. Now place the below ComboBox code in the `src/App.tsx`.

[Class-component]

```
`ts
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
  public render() {
    return (
      // specifies the tag for render the ComboBox component
      <ComboBoxComponent id='comboelement'/>
    );
  }
}
ReactDOM.render(<App />, document.getElementById('sample'));
`
```

[Functional-component]

```
`ts
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  return (
    // specifies the tag for render the ComboBox component

```

```
<ComboBoxComponent id='comboelement'/>
);
}
ReactDOM.render(<App />, document.getElementById('sample'));
`
```

Adding CSS reference

Import the ComboBox component required CSS references as follows in `src/App.css`.

```
`css
/ import the ComboBox dependency styles /
@import "../node_modules/@syncfusion/ej2-base/styles/material.css";
@import "../node_modules/@syncfusion/ej2-react-inputs/styles/material.css";
@import "../node_modules/@syncfusion/ej2-react-dropdowns/styles/material.css";
`
```

Binding data source

After initialization, populate the ComboBox with data using the `dataSource` property. Here, an array of string values is passed to the ComboBox component.

[Class-component]

```
`ts
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
// define the array of data
private sportsData: string[] = ['Badminton', 'Cricket', 'Football', 'Golf', 'Tennis'];
public render() {
return (
// specifies the tag for render the ComboBox component
<ComboBoxComponent id="comboelement" dataSource={this.sportsData} />
);
}
}
ReactDOM.render(<App />, document.getElementById('sample'));
`
```

[Functional-component]

```
`ts
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
// define the array of data
const sportsData: string[] = ['Badminton', 'Cricket', 'Football', 'Golf', 'Tennis'];
return (
// specifies the tag for render the ComboBox component
<ComboBoxComponent id="comboelement" dataSource={sportsData} />
);
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

Run the application

After completing the configuration required to render a basic ComboBox, run the following command to display the output in your default browser.

```
npm start
```

[Class-componnet]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
// define the array of data
sportsData = ['Badminton', 'Cricket', 'Football', 'Golf', 'Tennis'];
render() {
return (
// specifies the tag for render the ComboBox component
<ComboBoxComponent id="comboelement" dataSource={this.sportsData}
placeholder="Select a game"/>);
}
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
```

```
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}>, {}> {
  // define the array of data
  private sportsData: string[] = ['Badminton', 'Cricket', 'Football', 'Golf', 'Tennis'];
  public render() {
    return (
      // specifies the tag for render the ComboBox component
      <ComboBoxComponent id="comboelement" dataSource={this.sportsData}
placeholder="Select a game" />
    );
  }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

[Functional-componnet]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  // define the array of data
  const sportsData = ['Badminton', 'Cricket', 'Football', 'Golf', 'Tennis'];
  return (
    // specifies the tag for render the ComboBox component
    <ComboBoxComponent id="comboelement" dataSource={sportsData}
placeholder="Select a game"/>
  );
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  // define the array of data
  const sportsData: string[] = ['Badminton', 'Cricket', 'Football', 'Golf', 'Tennis'];
  return (
    // specifies the tag for render the ComboBox component
    <ComboBoxComponent id="comboelement" dataSource={sportsData}
placeholder="Select a game" />
  );
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

Custom values

The ComboBox allows the user to give input as custom value which is not required to present in predefined set of values. By default, this support is enabled by [allowCustom](#) property. In this case, both

text field and value field considered as same. The custom value will be sent to post back handler when a form is about to be submitted.

[Class-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    // defined the array of data
    sportsData = [
        { Id: 'game1', Game: 'Badminton' },
        { Id: 'game2', Game: 'Football' },
        { Id: 'game3', Game: 'Tennis' }
    ];
    // maps the appropriate column to fields property
    fields = { text: 'Game', value: 'Id' };
    render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement" fields={this.fields}
            dataSource={this.sportsData} allowCustom={true} placeholder="Select a
            game"/>);
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // defined the array of data
    private sportsData: { [key: string]: Object }[] = [
        { Id: 'game1', Game: 'Badminton' },
        { Id: 'game2', Game: 'Football' },
        { Id: 'game3', Game: 'Tennis' }
    ];
    // maps the appropriate column to fields property
    private fields: object = { text: 'Game', value: 'Id' };
    public render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement" fields={this.fields}
            dataSource={this.sportsData} allowCustom={true} placeholder="Select a game"
            />
        );
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

[Functional-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // defined the array of data
    const sportsData = [
        { Id: 'game1', Game: 'Badminton' },
        { Id: 'game2', Game: 'Football' },
        { Id: 'game3', Game: 'Tennis' }
    ];
    // maps the appropriate column to fields property
    const fields = { text: 'Game', value: 'Id' };
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" fields={fields}
        dataSource={sportsData} allowCustom={true} placeholder="Select a game"/>
    );
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // defined the array of data
    const sportsData: { [key: string]: Object }[] = [
        { Id: 'game1', Game: 'Badminton' },
        { Id: 'game2', Game: 'Football' },
        { Id: 'game3', Game: 'Tennis' }
    ];
    // maps the appropriate column to fields property
    const fields: object = { text: 'Game', value: 'Id' };
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" fields={fields}
        dataSource={sportsData} allowCustom={true} placeholder="Select a game" />
    );
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

Configure the popup list

By default, the width of the popup list automatically adjusts according to the ComboBox input element's width, and the height of the popup list has '300px'.

The height and width of the popup list can also be customized using the [popupHeight](#) and [popupWidth](#) property respectively.

In the following sample, popup list's width and height have configured.

[Class-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    // define the array of data
    sportsData = ['Badminton', 'Basketball', 'Cricket', 'Football', 'Golf',
    'Hockey', 'Rugby', 'Snooker', 'Tennis'];
    render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement" dataSource={this.sportsData}
            popupHeight="200px" popupWidth="250px" placeholder="select a game"/>);
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // define the array of data
    private sportsData: string[] = ['Badminton', 'Basketball', 'Cricket',
    'Football', 'Golf', 'Hockey', 'Rugby', 'Snooker', 'Tennis'];
    public render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement" dataSource={this.sportsData}
            popupHeight="200px" popupWidth="250px" placeholder="select a game" />
        );
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

[Functional-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // define the array of data
    const sportsData = ['Badminton', 'Basketball', 'Cricket', 'Football',
    'Golf', 'Hockey', 'Rugby', 'Snooker', 'Tennis'];
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" dataSource={sportsData}
        popupHeight="200px" popupWidth="250px" placeholder="select a game"/>);
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX


```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // define the array of data
    const sportsData: string[] = ['Badminton', 'Basketball', 'Cricket',
    'Football', 'Golf', 'Hockey', 'Rugby', 'Snooker', 'Tennis'];
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" dataSource={sportsData}
        popupHeight="200px" popupWidth="250px" placeholder="select a game" />
    );
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

See Also

- [How to bind the data](#)

Data binding in React Combo box component

The ComboBox loads the data either from local data sources or remote data services using the [dataSource](#) property. It supports the data type of `array` or `DataManager`.

The ComboBox also supports different kinds of data services such as OData, OData V4, and Web API, and data formats such as XML, JSON, and JSONP with the help of `DataManager` adaptors.

Fields	Type	Description
text	string	Specifies the display text of each list item.
value	number or string	Specifies the hidden data value which mapped to each list item that should be unique.
groupBy	string	Specifies the category under which the list item needs to be grouped.
iconCss	string	Specifies the icon class of each list item.

When binding complex data to the ComboBox, fields should be mapped correctly. Otherwise, the selected item remains undefined.

Binding local data

Local data can be represented in two ways as described below.

1. Array of simple data

The ComboBox has support to load array of primitive data such as strings and numbers. Here, both value and text field act the same.

[Class-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
```

```
export default class App extends React.Component {
  // define the array of string
  sportsData = ['Badminton', 'Cricket', 'Football', 'Golf', 'Tennis'];
  render() {
    return (
      // specifies the tag for render the ComboBox component
      <ComboBoxComponent id="comboelement" dataSource={this.sportsData}
placeholder="Select a game"/>);
    }
  }
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
  // define the array of string
  private sportsData: string[] = ['Badminton', 'Cricket', 'Football',
'Golf', 'Tennis'];
  public render() {
    return (
      // specifies the tag for render the ComboBox component
      <ComboBoxComponent id="comboelement" dataSource={this.sportsData}
placeholder="Select a game" />
    );
  }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

[Functional-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  // define the array of string
  const sportsData = ['Badminton', 'Cricket', 'Football', 'Golf',
'Tennis'];
  return (
    // specifies the tag for render the ComboBox component
    <ComboBoxComponent id="comboelement" dataSource={sportsData}
placeholder="Select a game"/>);
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
```

```
// define the array of string
const sportsData: string[] = ['Badminton', 'Cricket', 'Football', 'Golf', 'Tennis'];
return (
  // specifies the tag for render the ComboBox component
  <ComboBoxComponent id="comboelement" dataSource={sportsData}
placeholder="Select a game" />
);
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

2. Array of JSON data

The ComboBox can generate its list items through an array of complex data. For this, the appropriate columns should be mapped to the [fields](#) property.

In the following example, **Id** column and **Game** column from complex data have been mapped to the **value** field and **text** field, respectively.

[Class-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  // define the JSON of data
  sportsData = [
    { Id: 'game1', Game: 'Badminton' },
    { Id: 'game2', Game: 'Football' },
    { Id: 'game3', Game: 'Tennis' }
  ];
  // maps the appropriate column to fields property
  fields = { text: 'Game', value: 'Id' };
  render() {
    return (
      // specifies the tag for render the ComboBox component
      <ComboBoxComponent id="comboelement" dataSource={this.sportsData}
fields={this.fields} placeholder="Select a game"/>
    );
  }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
  // define the JSON of data
  private sportsData: { [key: string]: Object }[] = [
    { Id: 'game1', Game: 'Badminton' },
    { Id: 'game2', Game: 'Football' },
    { Id: 'game3', Game: 'Tennis' }
  ];
  // maps the appropriate column to fields property
```

```
private fields: object = { text: 'Game', value: 'Id' };
public render() {
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" dataSource={this.sportsData}
fields={this.fields} placeholder="Select a game" />
    );
}
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

[Functional-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // define the JSON of data
    const sportsData = [
        { Id: 'game1', Game: 'Badminton' },
        { Id: 'game2', Game: 'Football' },
        { Id: 'game3', Game: 'Tennis' }
    ];
    // maps the appropriate column to fields property
    const fields = { text: 'Game', value: 'Id' };
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" dataSource={sportsData}
fields={fields} placeholder="Select a game"/>
    );
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // define the JSON of data
    const sportsData: { [key: string]: Object }[] = [
        { Id: 'game1', Game: 'Badminton' },
        { Id: 'game2', Game: 'Football' },
        { Id: 'game3', Game: 'Tennis' }
    ];
    // maps the appropriate column to fields property
    const fields: object = { text: 'Game', value: 'Id' };
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" dataSource={sportsData}
fields={fields} placeholder="Select a game" />
    );
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

3. Array of Complex data

The ComboBox can generate its list items through an array of complex data. For this, the appropriate columns should be mapped to the

[fields](#) property.

In the following example, `Code.Id` column and `Country.Name` column from complex data have been mapped to the `value` field and `text` field, respectively.

[Class-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    // define the JSON of data
    countriesData = [
        { Country: { Name: 'Australia' }, Code: { Id: 'AU' } },
        { Country: { Name: 'Bermuda' }, Code: { Id: 'BM' } },
        { Country: { Name: 'Canada' }, Code: { Id: 'CA' } },
        { Country: { Name: 'Cameroon' }, Code: { Id: 'CM' } },
        { Country: { Name: 'Denmark' }, Code: { Id: 'DK' } },
        { Country: { Name: 'France' }, Code: { Id: 'FR' } }
    ];
    // maps the appropriate column to fields property
    fields = { text: 'Country.Name', value: 'Code.Id' };
    render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement" dataSource={this.countriesData}
            fields={this.fields} placeholder="Select a country"/>);
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
{% raw %}
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // define the JSON of data
    private countriesData: { [key: string]: Object }[] = [
        { Country: { Name: 'Australia' }, Code: { Id: 'AU' } },
        { Country: { Name: 'Bermuda' }, Code: { Id: 'BM' } },
        { Country: { Name: 'Canada' }, Code: { Id: 'CA' } },
        { Country: { Name: 'Cameroon' }, Code: { Id: 'CM' } },
        { Country: { Name: 'Denmark' }, Code: { Id: 'DK' } },
        { Country: { Name: 'France' }, Code: { Id: 'FR' } }
    ];
    // maps the appropriate column to fields property
    private fields: object = { text: 'Country.Name', value: 'Code.Id' };
    public render() {
        return (
```

```

        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement"
        dataSource={this.countriesData} fields={this.fields} placeholder="Select a
        country" />
      );
    }
  }
  ReactDOM.render(<App />, document.getElementById('sample'));
  {% endraw %}

```

[Functional-component]

INDEX.JSX

```

import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  // define the JSON of data
  const countriesData = [
    { Country: { Name: 'Australia' }, Code: { Id: 'AU' } },
    { Country: { Name: 'Bermuda' }, Code: { Id: 'BM' } },
    { Country: { Name: 'Canada' }, Code: { Id: 'CA' } },
    { Country: { Name: 'Cameroon' }, Code: { Id: 'CM' } },
    { Country: { Name: 'Denmark' }, Code: { Id: 'DK' } },
    { Country: { Name: 'France' }, Code: { Id: 'FR' } }
  ];
  // maps the appropriate column to fields property
  const fields = { text: 'Country.Name', value: 'Code.Id' };
  return (
    // specifies the tag for render the ComboBox component
    <ComboBoxComponent id="comboelement" dataSource={countriesData}
    fields={fields} placeholder="Select a country"/>);
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

INDEX.TSX

```

{% raw %}
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  // define the JSON of data
  const countriesData: { [key: string]: Object }[] = [
    { Country: { Name: 'Australia' }, Code: { Id: 'AU' } },
    { Country: { Name: 'Bermuda' }, Code: { Id: 'BM' } },
    { Country: { Name: 'Canada' }, Code: { Id: 'CA' } },
    { Country: { Name: 'Cameroon' }, Code: { Id: 'CM' } },
    { Country: { Name: 'Denmark' }, Code: { Id: 'DK' } },
    { Country: { Name: 'France' }, Code: { Id: 'FR' } }
  ];
  // maps the appropriate column to fields property
  const fields: object = { text: 'Country.Name', value: 'Code.Id' };
  return (
    // specifies the tag for render the ComboBox component

```

```
<ComboBoxComponent id="comboelement" dataSource={countriesData}
fields={fields} placeholder="Select a country" />
);
}
ReactDOM.render(<App />, document.getElementById('sample'));
{% endraw %}
```

Binding remote data

The ComboBox supports retrieval of data from remote data services with the help of **DataManager** component. The [Query](#) property is used to fetch data from the database and bind it to the ComboBox.

The following sample displays the first 6 contacts from “Customers” table of the **Northwind** Data Service.

[Class-component]

INDEX.JSX

```
// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  // bind the DataManager instance to dataSource property
  customerData = new DataManager({
    adaptor: new ODataV4Adaptor,
    crossDomain: true,
    url: 'https://services.odata.org/V4/Northwind/Northwind.svc/'
  });
  // bind the Query instance to query property
  query = new Query().from('Customers').select(['ContactName',
'CustomerID']).take(6);
  // maps the appropriate column to fields property
  fields = { text: 'ContactName', value: 'CustomerID' };
  // sort the resulted items
  sortOrder = 'Ascending';
  render() {
    return (
      // specifies the tag for render the ComboBox component
      <ComboBoxComponent id="comboelement" query={this.query}
dataSource={this.customerData} fields={this.fields} placeholder="Select a
customer" sortOrder={this.sortOrder}/>);
    }
  }
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { SortOrder } from '@syncfusion/ej2-lists';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
```

```
export default class App extends React.Component<{}, {}> {
  // bind the DataManager instance to dataSource property
  private customerData: DataManager = new DataManager({
    adaptor: new ODataV4Adaptor,
    crossDomain: true,
    url: 'https://services.odata.org/V4/Northwind/Northwind.svc/'
  });
  // bind the Query instance to query property
  private query: Query = new
  Query().from('Customers').select(['ContactName', 'CustomerID']).take(6);
  // maps the appropriate column to fields property
  private fields: object = { text: 'ContactName', value: 'CustomerID' };
  // sort the resulted items
  private sortOrder: SortOrder = 'Ascending';
  public render() {
    return (
      // specifies the tag for render the ComboBox component
      <ComboBoxComponent id="comboelement" query={this.query}
dataSource={this.customerData}
fields={this.fields} placeholder="Select a customer"
sortOrder={this.sortOrder} />
    );
  }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

[Functional-component]

INDEX.JSX

```
// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  // bind the DataManager instance to dataSource property
  const customerData = new DataManager({
    adaptor: new ODataV4Adaptor,
    crossDomain: true,
    url: 'https://services.odata.org/V4/Northwind/Northwind.svc/'
  });
  // bind the Query instance to query property
  const query = new Query().from('Customers').select(['ContactName',
'CustomerID']).take(6);
  // maps the appropriate column to fields property
  const fields = { text: 'ContactName', value: 'CustomerID' };
  // sort the resulted items
  const sortOrder = 'Ascending';
  return (
    // specifies the tag for render the ComboBox component
    <ComboBoxComponent id="comboelement" query={query}
dataSource={customerData} fields={fields} placeholder="Select a customer"
sortOrder={sortOrder}/>
  );
}
ReactDOM.render(<App />, document.getElementById('sample'));
```


INDEX.TSX

```
// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { SortOrder } from '@syncfusion/ej2-lists';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // bind the DataManager instance to dataSource property
    const customerData: DataManager = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url: 'https://services.odata.org/V4/Northwind/Northwind.svc/'
    });
    // bind the Query instance to query property
    const query: Query = new Query().from('Customers').select(['ContactName',
    'CustomerID']).take(6);
    // maps the appropriate column to fields property
    const fields: object = { text: 'ContactName', value: 'CustomerID' };
    // sort the resulted items
    const sortOrder: SortOrder = 'Ascending';
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" query={query}
        dataSource={customerData}
        fields={fields} placeholder="Select a customer"
        sortOrder={sortOrder} />
    );
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

See Also

- [How to achieve cascading](#)
- [How to load data using template](#)
- [How to group the data using header](#)
- [How to filter the bound data](#)

Value binding in ComboBox Component

Value binding in the ComboBox control allows you to associate data values with each list item. This facilitates managing and retrieving selected values efficiently. The ComboBox component provides flexibility in binding both primitive data types and complex objects.

Primitive Data Types

The ComboBox control provides flexible binding capabilities for primitive data types like strings and numbers. You can effortlessly bind local primitive data arrays, fetch and bind data from remote sources, and even custom data binding to suit specific requirements. Bind the value of primitive data to the [value](#) property of the ComboBox.

Primitive data types include:

- String
- Number
- Boolean
- Null

The following sample shows the example for preselect values for primitive data type

[Class-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  constructor(props) {
    super(props);
    this.records = ["Item 1", "Item 2", "Item 3", "Item 4", "Item 5", "Item 6", "Item 7", "Item 8", "Item 9", "Item 10"];
  }
  fields = { text: 'text', value: 'id' };
  value = "Item 5";

  render() {
    return (
      // specifies the tag for render the ComboBox component
      <ComboBoxComponent id="datas" dataSource={this.records}
      value={this.value} placeholder="e.g. Item 1" allowFiltering={false}
      popupHeight="200px" >
        </ComboBoxComponent>;
    );
  }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
  // maps the appropriate column to fields property
  private fields: object = { text: 'text', value: 'id' };
  // define the array of string
  private records: string[] = [];
  private value : string = "Item 1";
  constructor(props) {
    super(props);
    this.records = ["Item 1", "Item 2", "Item 3", "Item 4", "Item 5", "Item 6", "Item 7", "Item 8", "Item 9", "Item 10"];
  }
  public render() {
    return (
      // specifies the tag for render the ComboBox component
      <ComboBoxComponent id="datas" dataSource={this.records}
      value={this.value} placeholder="e.g. Item 1" allowObjectBinding={true}
      allowFiltering={false} popupHeight="200px" >
    );
  }
}
```

```

        </ComboBoxComponent>
    );
}
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

Object Data Types

In the ComboBox control, object binding allows you to bind to a dataset of objects. When [allowObjectBinding](#) is enabled, the value of the control will be an object of the same type as the selected item in the [value](#) property. This feature seamlessly binds arrays of objects, whether sourced locally, retrieved from remote endpoints, or customized to suit specific application needs.

The following sample shows the example for preselect values for object data type

[Link to the Video](#)

INDEX.JSX

```

import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    constructor(props) {
        super(props);
        this.records = Array.from({ length: 150 }, (_, i) => ({
            id: 'id' + (i + 1),
            text: `Item ${i + 1}`,
        }));
    }
    fields = { text: 'text', value: 'id' };
    value = { text: 'Item 5', value: 'id5' };

    render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="datas" dataSource={this.records}
            value={this.value} fields={this.fields} placeholder="e.g. Item 1"
            allowObjectBinding={true} allowFiltering={false} popupHeight="200px" >
                </ComboBoxComponent>);
        )
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

INDEX.TSX

```

import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // maps the appropriate column to fields property
    private fields: object = { text: 'text', value: 'id' };
    // define the array of string
    private records: { [key: string]: Object }[] = [];
    private value : { [key: string]: Object } = { id: 'id1', text: 'Item 1'
};

```

```

    constructor(props) {
      super(props);
      this.records = Array.from({ length: 150 }, (_, i) => ({
        id: 'id' + (i + 1),
        text: `Item ${i + 1}`,
      }));
    }
    public render() {
      return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="datas" dataSource={this.records}
        value={this.value} fields={this.fields} placeholder="e.g. Item 1"
        allowObjectBinding={true} allowFiltering={false} popupHeight="200px" >
          </ComboBoxComponent>
        );
      }
    }
  }
ReactDOM.render(<App />, document.getElementById('sample'));

```

Templates in React Combo box component

The ComboBox has been provided with several options to customize each list items, group title, header, and footer elements.

To get started with React ComboBox templates, you can check on this video:

Item template

The content of each list item within the ComboBox can be customized with the help of [itemTemplate](#) property.

In the following sample, each list item is split into two columns to display relevant data's.

[Class-component]

INDEX.JSX

```

// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  // bind the DataManager instance to dataSource property
  employeeData = new DataManager({
    adaptor: new ODataV4Adaptor,
    crossDomain: true,
    url: 'https://services.odata.org/V4/Northwind/Northwind.svc/'
  });
  // bind the Query instance to query property
  query = new Query().from('Employees').select(['FirstName', 'City',
  'EmployeeID']).take(6);
  // maps the appropriate column to fields property
  fields = { text: 'FirstName', value: 'EmployeeID' };
  // sort the resulted items
  sortOrder = 'Ascending';
  // set the value to itemTemplate property
  itemTemplate(data) {

```

```

        return (<span className='item'><span
className='name'>{data.FirstName}</span><span
className='city'>{data.City}</span></span>);
    }
    render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement" query={this.query}
itemTemplate={this.itemTemplate} dataSource={this.employeeData}
fields={this.fields} sortOrder={this.sortOrder} placeholder="Select an
employee"/>);
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

INDEX.TSX

```

// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { SortOrder } from '@syncfusion/ej2-lists';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // bind the DataManager instance to dataSource property
    private employeeData: DataManager = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url: 'https://services.odata.org/V4/Northwind/Northwind.svc/'
    });
    // bind the Query instance to query property
    private query: Query = new Query().from('Employees').select(['FirstName',
'City', 'EmployeeID']).take(6);
    // maps the appropriate column to fields property
    private fields: object = { text: 'FirstName', value: 'EmployeeID' };
    // sort the resulted items
    private sortOrder: SortOrder = 'Ascending';
    // set the value to itemTemplate property
    public itemTemplate(data: any): JSX.Element {
        return (
            <span className='item' ><span
className='name'>{data.FirstName}</span><span
className='city'>{data.City}</span></span>
        );
    }
    public render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement" query={this.query}
itemTemplate={this.itemTemplate} dataSource={this.employeeData}
fields={this.fields} sortOrder={this.sortOrder} placeholder="Select an
employee" />
        );
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

[Functional-component]

INDEX.JSX

```
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // bind the DataManager instance to dataSource property
    const employeeData = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url: 'https://services.odata.org/V4/Northwind/Northwind.svc/'
    });
    // bind the Query instance to query property
    const query = new Query().from('Employees').select(['FirstName', 'City', 'EmployeeID']).take(6);
    // maps the appropriate column to fields property
    const fields = { text: 'FirstName', value: 'EmployeeID' };
    // sort the resulted items
    const sortOrder = 'Ascending';
    // set the value to itemTemplate property
    function itemTemplate(data) {
        return (<span className='item'><span
        className='name'>{data.FirstName}</span><span
        className='city'>{data.City}</span></span>);
    }
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" query={query}
        itemTemplate={itemTemplate} dataSource={employeeData} fields={fields}
        sortOrder={sortOrder} placeholder="Select an employee"/>);
    }
    ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { SortOrder } from '@syncfusion/ej2-lists';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // bind the DataManager instance to dataSource property
    const employeeData: DataManager = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url: 'https://services.odata.org/V4/Northwind/Northwind.svc/'
    });
    // bind the Query instance to query property
    const query: Query = new Query().from('Employees').select(['FirstName', 'City', 'EmployeeID']).take(6);
    // maps the appropriate column to fields property
    const fields: object = { text: 'FirstName', value: 'EmployeeID' };
```

```
// sort the resulted items
const sortOrder: SortOrder = 'Ascending';
// set the value to itemTemplate property
function itemTemplate(data: any): JSX.Element {
    return (
        <span className='item' ><span
className='name'>{data.FirstName}</span><span
className='city'>{data.City}</span></span>
    );
}
return (
    // specifies the tag for render the ComboBox component
    <ComboBoxComponent id="comboelement" query={query}
itemTemplate={itemTemplate} dataSource={employeeData} fields={fields}
sortOrder={sortOrder} placeholder="Select an employee" />
);
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

Group template

The group header title under which appropriate sub-items are categorized can also be customize with the help of [groupTemplate](#) property. This template is common for both inline and floating group header template.

In the following sample, employees are grouped according to their city.

[Class-component]

INDEX.JSX

```
// import DataManager related classes
import { DataManager, ODataV4Adaptor, Predicate, Query } from
'@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    // bind the DataManager instance to dataSource property
    employeeData = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url: 'https://services.odata.org/V4/Northwind/Northwind.svc/'
    });
    // form predicate to fetch the grouped data
    groupPredicate = new Predicate('City', 'equal', 'london').or('City',
'equal', 'seattle');
    // bind the Query instance to query property
    query = new Query().from('Employees').select(['FirstName', 'City',
'EmployeeID']).take(6).where(this.groupPredicate);
    // maps the appropriate column to fields property
    fields = { text: 'FirstName', value: 'EmployeeID', groupBy: 'City' };
    // sort the resulted items
    sortOrder = 'Ascending';
    // set the value to groupTemplate
    groupTemplate(data) {
        return <strong>{data.City}</strong>;
    }
}
```

```

    }
    render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement" query={this.query}
groupTemplate={this.groupTemplate} dataSource={this.employeeData}
fields={this.fields} sortOrder={this.sortOrder} placeholder="Select an
employee"/>);
        }
    }
    ReactDOM.render(<App />, document.getElementById('sample'));

```

INDEX.TSX

```

// import DataManager related classes
import { DataManager, ODataV4Adaptor, Predicate, Query } from
'@syncfusion/ej2-data';
import { SortOrder } from '@syncfusion/ej2-lists';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // bind the DataManager instance to dataSource property
    private employeeData: DataManager = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url: 'https://services.odata.org/V4/Northwind/Northwind.svc/'
    });
    // form predicate to fetch the grouped data
    private groupPredicate = new Predicate('City', 'equal',
'london').or('City', 'equal', 'seattle');
    // bind the Query instance to query property
    private query: Query = new Query().from('Employees').select(['FirstName',
'City', 'EmployeeID']).take(6).where(this.groupPredicate);
    // maps the appropriate column to fields property
    private fields: object = { text: 'FirstName', value: 'EmployeeID',
groupBy: 'City' };
    // sort the resulted items
    private sortOrder: SortOrder = 'Ascending';
    // set the value to groupTemplate
    public groupTemplate(data: any): JSX.Element {
        return (
            <strong>{data.City}</strong>
        );
    }
    public render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement" query={this.query}
groupTemplate={this.groupTemplate} dataSource={this.employeeData}
fields={this.fields} sortOrder={this.sortOrder} placeholder="Select an
employee" />
        );
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));

```


[Functional-component]

INDEX.JSX

```
import { DataManager, ODataV4Adaptor, Predicate, Query } from
 '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // bind the DataManager instance to dataSource property
    const employeeData = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url: 'https://services.odata.org/V4/Northwind/Northwind.svc/'
    });
    // form predicate to fetch the grouped data
    const groupPredicate = new Predicate('City', 'equal',
    'london').or('City', 'equal', 'seattle');
    // bind the Query instance to query property
    const query = new Query().from('Employees').select(['FirstName', 'City',
    'EmployeeID']).take(6).where(groupPredicate);
    // maps the appropriate column to fields property
    const fields = { text: 'FirstName', value: 'EmployeeID', groupBy: 'City'
    };
    // sort the resulted items
    const sortOrder = 'Ascending';
    // set the value to groupTemplate
    function groupTemplate(data) {
        return (<strong>{data.City}</strong>);
    }
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" query={query}
        groupTemplate={groupTemplate} dataSource={employeeData} fields={fields}
        sortOrder={sortOrder} placeholder="Select an employee"/>);
    }
    ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { DataManager, ODataV4Adaptor, Predicate, Query } from
 '@syncfusion/ej2-data';
import { SortOrder } from '@syncfusion/ej2-lists';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // bind the DataManager instance to dataSource property
    const employeeData: DataManager = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url: 'https://services.odata.org/V4/Northwind/Northwind.svc/'
    });
    // form predicate to fetch the grouped data
```

```
const groupPredicate = new Predicate('City', 'equal',
'london').or('City', 'equal', 'seattle');
// bind the Query instance to query property
const query: Query = new Query().from('Employees').select(['FirstName',
'City', 'EmployeeID']).take(6).where(groupPredicate);
// maps the appropriate column to fields property
const fields: object = { text: 'FirstName', value: 'EmployeeID', groupBy:
'City' };
// sort the resulted items
const sortOrder: SortOrder = 'Ascending';
// set the value to groupTemplate
function groupTemplate(data: any): JSX.Element {
    return (
        <strong>{data.City}</strong>
    );
}
return (
    // specifies the tag for render the ComboBox component
    <ComboBoxComponent id="comboelement" query={query}
groupTemplate={groupTemplate} dataSource={employeeData} fields={fields}
sortOrder={sortOrder} placeholder="Select an employee" />
);
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

Header template

The header element is shown statically at the top of the popup list items within the ComboBox, and any custom element can be placed as a header element using the

[headerTemplate](#) property.

In the following sample, the list items and its headers are designed and displayed as two columns similar to multiple columns of the grid.

[Class-component]

INDEX.JSX

```
// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    // bind the DataManager instance to dataSource property
    employeeData = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url: 'https://services.odata.org/V4/Northwind/Northwind.svc/'
    });
    // bind the Query instance to query property
    query = new Query().from('Employees').select(['FirstName', 'City',
'EmployeeID']).take(6);
    // maps the appropriate column to fields property
    fields = { text: 'FirstName', value: 'EmployeeID' };
    // sort the resulted items
```

```

        sortOrder = 'Ascending';
        // set the value to header template
        headerTemplate(data) {
            return (<span className='head'><span
className='name'>Name</span><span className='city'>City</span></span>);
        }
        // set the value to item template
        itemTemplate(data) {
            return (<span className='item'><span
className='name'>{data.FirstName}</span><span
className='city'>{data.City}</span></span>);
        }
        render() {
            return (
                // specifies the tag for render the ComboBox component
                <ComboBoxComponent id="comboelement" query={this.query}
headerTemplate={this.headerTemplate} dataSource={this.employeeData}
sortOrder={this.sortOrder} itemTemplate={this.itemTemplate}
fields={this.fields} placeholder="Select an employee"/>);
        }
    }
    ReactDOM.render(<App />, document.getElementById('sample'));

```

INDEX.TSX

```

// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { SortOrder } from '@syncfusion/ej2-lists';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // bind the DataManager instance to dataSource property
    private employeeData: DataManager = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url: 'https://services.odata.org/V4/Northwind/Northwind.svc/'
    });
    // bind the Query instance to query property
    private query: Query = new Query().from('Employees').select(['FirstName',
'City', 'EmployeeID']).take(6);
    // maps the appropriate column to fields property
    private fields: object = { text: 'FirstName', value: 'EmployeeID' };
    // sort the resulted items
    private sortOrder: SortOrder = 'Ascending';
    // set the value to header template
    public headerTemplate(data: any): JSX.Element {
        return (
            <span className='head'><span className='name'>Name</span><span
className='city'>City</span></span>
        );
    }
    // set the value to item template
    public itemTemplate(data: any): JSX.Element {
        return (

```

```

        <span className='item' ><span
        className='name'>{data.FirstName}</span><span
        className='city'>{data.City}</span></span></span>
    );
}
    public render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement" query={this.query}
            headerTemplate={this.headerTemplate} dataSource={this.employeeData}
            sortOrder={this.sortOrder} itemTemplate={this.itemTemplate}
            fields={this.fields} placeholder="Select an employee" />
        );
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

[Functional-component]

INDEX.JSX

```

// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // bind the DataManager instance to dataSource property
    const employeeData = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url: 'https://services.odata.org/V4/Northwind/Northwind.svc/'
    });
    // bind the Query instance to query property
    const query = new Query().from('Employees').select(['FirstName', 'City',
    'EmployeeID']).take(6);
    // maps the appropriate column to fields property
    const fields = { text: 'FirstName', value: 'EmployeeID' };
    // sort the resulted items
    const sortOrder = 'Ascending';
    // set the value to header template
    function headerTemplate(data) {
        return (<span className='head'><span
        className='name'>Name</span><span className='city'>City</span></span>);
    }
    // set the value to item template
    function itemTemplate(data) {
        return (<span className='item'><span
        className='name'>{data.FirstName}</span><span
        className='city'>{data.City}</span></span>);
    }
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" query={query}
        headerTemplate={headerTemplate} dataSource={employeeData}

```

```
sortOrder={sortOrder} itemTemplate={itemTemplate} fields={fields}
placeholder="Select an employee"/>);
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { SortOrder } from '@syncfusion/ej2-lists';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // bind the DataManager instance to dataSource property
    const employeeData: DataManager = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url: 'https://services.odata.org/V4/Northwind/Northwind.svc/'
    });
    // bind the Query instance to query property
    const query: Query = new Query().from('Employees').select(['FirstName',
    'City', 'EmployeeID']).take(6);
    // maps the appropriate column to fields property
    const fields: object = { text: 'FirstName', value: 'EmployeeID' };
    // sort the resulted items
    const sortOrder: SortOrder = 'Ascending';
    // set the value to header template
    function headerTemplate(data: any): JSX.Element {
        return (
            <span className='head'><span className='name'>Name</span><span
            className='city'>City</span></span>
        );
    }
    // set the value to item template
    function itemTemplate(data: any): JSX.Element {
        return (
            <span className='item' ><span
            className='name'>{data.FirstName}</span><span
            className='city'>{data.City}</span></span>
        );
    }
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" query={query}
        headerTemplate={headerTemplate} dataSource={employeeData}
        sortOrder={sortOrder} itemTemplate={itemTemplate} fields={fields}
        placeholder="Select an employee" />
    );
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

Footer template

The ComboBox has options to show a footer element at the bottom of the list items in the popup list. Here, you can place any custom element as a footer element using the [footerTemplate](#) property.

In the following sample, footer element displays the total number of list items present in the ComboBox.

[Class-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    // define the array of data
    sportsData = ["BasketBall", "Cricket", "Football", "Golf"];
    // set the value to footer template
    footerTemplate(data) {
        return (<span className='foot' />);
    }
    render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement"
            footerTemplate={this.footerTemplate} dataSource={this.sportsData}
            placeholder="Select a game" />);
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // define the array of data
    private sportsData: string[] = ["BasketBall", "Cricket", "Football",
    "Golf"];
    // set the value to footer template
    public footerTemplate(data: any): JSX.Element {
        return (
            <span className='foot' />
        );
    }
    public render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement"
            footerTemplate={this.footerTemplate} dataSource={this.sportsData}
            placeholder="Select a game" />
        );
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

[Functional-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
```

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  // define the array of data
  const sportsData = ["BasketBall", "Cricket", "Football", "Golf"];
  // set the value to footer template
  function footerTemplate(data) {
    return (<span className='foot' />);
  }
  return (
    // specifies the tag for render the ComboBox component
    <ComboBoxComponent id="comboelement" footerTemplate={footerTemplate}
    dataSource={sportsData} placeholder="Select a game"/>);
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  // define the array of data
  const sportsData: string[] = ["BasketBall", "Cricket", "Football",
  "Golf"];
  // set the value to footer template
  function footerTemplate(data: any): JSX.Element {
    return (
      <span className='foot' />
    );
  }
  return (
    // specifies the tag for render the ComboBox component
    <ComboBoxComponent id="comboelement" footerTemplate={footerTemplate}
    dataSource={sportsData} placeholder="Select a game" />
  );
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

No records template

The ComboBox is provided with support to custom design the popup list content when no data is found and no matches found on search with the help of

[noRecordsTemplate](#) property.

In the following sample, popup list content displays the notification of no data available.

[Class-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  // define the array of data
```

```

data = [];
// set the value to noRecords template
noRecordsTemplate(data) {
    return (<span className='norecord'> NO DATA AVAILABLE</span>);
}
render() {
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement"
noRecordsTemplate={this.noRecordsTemplate =
this.noRecordsTemplate.bind(this)} dataSource={this.data} placeholder="Select
an item"/>);
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

INDEX.TSX

```

import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // define the array of data
    private data: string[] = [];
    // set the value to noRecords template
    public noRecordsTemplate(data: any): JSX.Element {
        return (
            <span className='norecord'> NO DATA AVAILABLE</span>
        );
    }
    public render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement"
noRecordsTemplate={this.noRecordsTemplate =
this.noRecordsTemplate.bind(this)} dataSource={this.data} placeholder="Select
an item" />
        );
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

[Functional-component]

INDEX.JSX

```

import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // define the array of data
    const data = [];
    // set the value to noRecords template
    function noRecordsTemplate(data) {
        return (<span className='norecord'> NO DATA AVAILABLE</span>);
    }
}

```



```

    return (
      // specifies the tag for render the ComboBox component
      <ComboBoxComponent id="comboelement" noRecordsTemplate={noRecordsTemplate}
      = noRecordsTemplate.bind(this) dataSource={data} placeholder="Select an
      item"/>);
    }
    ReactDOM.render(<App />, document.getElementById('sample'));

```

INDEX.TSX

```

import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  // define the array of data
  const data: string[] = [];
  // set the value to noRecords template
  function noRecordsTemplate(data: any): JSX.Element {
    return (
      <span className='norecord'> NO DATA AVAILABLE</span>
    );
  }
  return (
    // specifies the tag for render the ComboBox component
    <ComboBoxComponent id="comboelement"
    noRecordsTemplate={noRecordsTemplate = noRecordsTemplate.bind(this)}
    dataSource={data} placeholder="Select an item" />
  );
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

Action failure template

There is also an option to custom design the popup list content when the data fetch request fails at the remote server. This can be achieved using the

[actionFailureTemplate](#) property.

In the following sample, when the data fetch request fails, the ComboBox displays the notification.

[Class-component]

INDEX.JSX

```

// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  // bind the DataManager instance to dataSource property
  customerData = new DataManager({
    adaptor: new ODataV4Adaptor,
    crossDomain: true,
    url: 'http://services.odata.org/V4/Northwind/Northwind.svc/'
  });
  // bind the Query instance to query property

```

```

    query = new Query().from('Customers').select(['ContactName',
'CustomerID']).take(6);
    // maps the appropriate column to fields property
    fields = { text: 'ContactName', value: 'CustomerID' };
    // set the value to action failure template
    template(data) {
        return (<span className='action-failure'> Data fetch get
fails</span>);
    }
    render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement"
actionFailureTemplate={this.template = this.template.bind(this)}
query={this.query} dataSource={this.customerData} fields={this.fields}
placeholder="Select a customer"/>);
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

INDEX.TSX

```

// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // bind the DataManager instance to dataSource property
    private customerData: DataManager = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url: 'http://services.odata.org/V4/Northwind/Northwind.svcs/'
    });
    // bind the Query instance to query property
    private query: Query = new
Query().from('Customers').select(['ContactName', 'CustomerID']).take(6);
    // maps the appropriate column to fields property
    private fields: object = { text: 'ContactName', value: 'CustomerID' };
    // set the value to action failure template
    public template(data: any): JSX.Element {
        return (
            <span className='action-failure'> Data fetch get fails</span>
        );
    }
    public render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement"
actionFailureTemplate={this.template = this.template.bind(this)}
query={this.query} dataSource={this.customerData} fields={this.fields}
placeholder="Select a customer" />
        );
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

[Functional-component]

INDEX.JSX

```
// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';

function App() {
    // bind the DataManager instance to dataSource property
    const customerData = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url: 'http://services.odata.org/V4/Northwind/Northwind.svc/'
    });
    // bind the Query instance to query property
    const query = new Query().from('Customers').select(['ContactName',
    'CustomerID']).take(6);
    // maps the appropriate column to fields property
    const fields = { text: 'ContactName', value: 'CustomerID' };
    // set the value to action failure template
    function template(data) {
        return (<span className='action-failure'> Data fetch get
    fails</span>);
    }
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" actionFailureTemplate={template =
    template.bind(this)} query={query} dataSource={customerData} fields={fields}
    placeholder="Select a customer"/>);
    )
    ReactDOM.render(<App />, document.getElementById('sample'));
}
```

INDEX.TSX

```
// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';

function App() {
    // bind the DataManager instance to dataSource property
    const customerData: DataManager = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url: 'http://services.odata.org/V4/Northwind/Northwind.svc/'
    });
    // bind the Query instance to query property
    const query: Query = new Query().from('Customers').select(['ContactName',
    'CustomerID']).take(6);
    // maps the appropriate column to fields property
    const fields: object = { text: 'ContactName', value: 'CustomerID' };
    // set the value to action failure template
    function template(data: any): JSX.Element {

```

```

        return (
            <span className='action-failure'> Data fetch get fails</span>
        );
    }
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" actionFailureTemplate={template
= template.bind(this) } query={query} dataSource={customerData}
fields={fields} placeholder="Select a customer" />
    );
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

See Also

- [How to achieve filtering](#)
- [How to group the data using header](#)
- [How to show the list items with icon](#)

Virtualization in ComboBox Component

ComboBox virtualization is a technique used to efficiently render extensive lists of items while minimizing the impact on performance. This method is particularly advantageous when dealing with large datasets because it ensures that only a fixed number of DOM (Document Object Model) elements are created. When scrolling through the list, existing DOM elements are reused to display relevant data instead of generating new elements for each item. This recycling process is managed internally.

During virtual scrolling, the data retrieved from the data source depends on the popup height and the calculation of the list item height. Enabling the [enableVirtualization](#) option in a ComboBox activates this virtualization technique.

When fetching data from the data source, the [actionBegin](#) event is triggered before data retrieval begins. Then, the [actionComplete](#) event is triggered once the data is successfully fetched.

Furthermore, Incremental Search is supported with virtualization in the Combobox component. When a key is typed, the focus is moved to the respective element in the open popup state. In the closed popup state, the popup opens, and focus is moved to the respective element in the popup list based on the typed key. The Incremental Search functionality is well-suited for scenarios involving remote data binding.

When the enableVirtualization property is enabled, the `skip` and `take` properties provided by the user within the Query class at the initial state or during the `actionBegin` or `actionComplete` events will not be considered, since it is internally managed and calculated based on certain dimensions with respect to the popup height.

Binding local data

The Combobox can generate its list items through an array of complex data. For this, the appropriate columns should be mapped to the [fields](#) property. When using virtual scrolling, the list updates based on the scroll offset value, triggering a request to fetch more data from the server. As the data is being fetched, the `actionBegin` event occurs before the data retrieval starts. Once the data retrieval is successful, the `actionComplete` event is triggered, indicating that the data fetch process is complete.

In the following example, **id** column and **text** column from complex data have been mapped to the **value** field and **text** field, respectively.

[Class-component]

INDEX.JSX

```
import { ComboBoxComponent, Inject, VirtualScroll } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  constructor(props) {
    super(props);
    this.records = Array.from({ length: 150 }, (_, i) => ({
      id: 'id' + (i + 1),
      text: `Item ${i + 1}`,
    }));
  }
  fields = { text: 'text', value: 'id' };

  render() {
    return (
      // specifies the tag for render the ComboBox component
      <ComboBoxComponent id="datas" dataSource={this.records}
fields={this.fields} placeholder="e.g. Item 1" enableVirtualization={true}
allowFiltering={false} popupHeight="200px" >
        <Inject services={[VirtualScroll]}/>
      </ComboBoxComponent>);
  }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent, Inject, VirtualScroll } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
  // maps the appropriate column to fields property
  private fields: object = { text: 'text', value: 'id' };
  // define the array of string
  private records: { [key: string]: Object }[] = [];
  constructor(props) {
    super(props);
    this.records = Array.from({ length: 150 }, (_, i) => ({
      id: 'id' + (i + 1),
      text: `Item ${i + 1}`,
    }));
  }
  public render() {
    return (
      // specifies the tag for render the ComboBox component

```

```

        <ComboBoxComponent id="datas" dataSource={this.records}
        fields={this.fields} placeholder="e.g. Item 1" enableVirtualization={true}
        allowFiltering={false} popupHeight="200px" >
            <Inject services={[VirtualScroll]} />
        </ComboBoxComponent>
    );
}
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

Binding remote data

The Combobox supports retrieval of data from remote data services with the help of **DataManager** component. When using remote data, it initially fetches all the data from the server, triggering the **actionBegin** and **actionComplete** events, and then stores the data locally. During virtual scrolling, additional data is retrieved from the locally stored data, triggering the **actionBegin** and **actionComplete** events at that time as well.

The following sample displays the OrderId from the **Orders** Data Service.

[Class-component]

INDEX.JSX

```

import { ComboBoxComponent, Inject, VirtualScroll } from '@syncfusion/ej2-react-dropdowns';
import { DataManager, WebApiAdaptor, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    customerData = new DataManager({
        url: 'https://services.syncfusion.com/react/production/api/Orders',
        adaptor: new WebApiAdaptor,
        crossDomain: true
    });
    customerField = { text: 'OrderID', value: 'OrderID' };

    render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="datas" dataSource={this.customerData}
            fields={this.customerField} placeholder="OrderID" enableVirtualization={true}
            allowFiltering={true} popupHeight="200px" >
                <Inject services={[VirtualScroll]} />
            </ComboBoxComponent>);
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

INDEX.TSX

```

import { ComboBoxComponent, Inject, VirtualScroll } from '@syncfusion/ej2-react-dropdowns';
import { DataManager, WebApiAdaptor, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import * as ReactDOM from 'react-dom';

```

```
export default class App extends React.Component<{}, {}> {
  // maps the appropriate column to fields property
  private customerField: object = { text: 'OrderID', value: 'OrderID' };

  private customerData: DataManager = new DataManager({
    url: 'https://services.syncfusion.com/react/production/api/Orders',
    adaptor: new WebApiAdaptor,
    crossDomain: true
  });

  public render() {
    return (
      // specifies the tag for render the ComboBox component
      <ComboBoxComponent id="datas" dataSource={this.customerData}
fields={this.customerField} placeholder="OrderID" enableVirtualization={true}
allowFiltering={true} popupHeight="200px" >
        <Inject services={[VirtualScroll]} />
      </ComboBoxComponent>
    );
  }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

Grouping

The Combobox component supports grouping with Virtualization. It allows you to organize elements into groups based on different categories. Each item in the list can be classified using the [groupBy](#) field in the data table. After grouping, virtualization works similarly to local data binding, providing a seamless user experience. When the data source is bound to remote data, an initial request is made to retrieve all data for the purpose of grouping. Subsequently, the grouped data works in the same way as local data binding virtualization, enhancing performance and responsiveness.

The following sample shows the example for Grouping with Virtualization.

[Class-component]

INDEX.JSX

```
import { ComboBoxComponent, Inject, VirtualScroll } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  constructor(props) {
    super(props);
    for (let i = 1; i <= 150; i++) {
      let item = {};
      item.id = 'id' + i;
      item.text = `Item ${i}`;
      // Generate a random number between 1 and 4 to determine the
group
      const randomGroup = Math.floor(Math.random() * 4) + 1;
      switch (randomGroup) {
        case 1:
          item.group = 'Group A';
          break;
        case 2:
          item.group = 'Group B';
```

```

                break;
            case 3:
                item.group = 'Group C';
                break;
            case 4:
                item.group = 'Group D';
                break;
            default:
                break;
        }
        this.records.push(item);
    }
}
fields = { groupBy: 'group', text: 'text', value: 'id' };

render() {
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="datas" dataSource={this.records}
fields={this.fields} placeholder="e.g. Item 1" enableVirtualization={true}
allowFiltering={true} popupHeight="200px" >
            <Inject services={[VirtualScroll]}/>
        </ComboBoxComponent>);
}
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

INDEX.TSX

```

import { ComboBoxComponent, Inject, VirtualScroll } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // maps the appropriate column to fields property
    private fields: object = { groupBy: 'group', text: 'text', value: 'id' };
    // define the array of string
    private records: { [key: string]: Object }[] = [];
    constructor(props) {
        super(props);
        for (let i = 1; i <= 150; i++) {
            const item: { id: string, text: string, group: string } = {
                id: 'id' + i,
                text: `Item ${i}`,
                group: ''
            };
            // Generate a random number between 1 and 4 to determine the
group
            const randomGroup = Math.floor(Math.random() * 4) + 1;
            switch (randomGroup) {
                case 1:
                    item.group = 'Group A';
                    break;
                case 2:
                    item.group = 'Group B';

```



```

        break;
      case 3:
        item.group = 'Group C';
        break;
      case 4:
        item.group = 'Group D';
        break;
      default:
        break;
    }
    this.records.push(item);
  }
}

public render() {
  return (
    // specifies the tag for render the ComboBox component
    <ComboBoxComponent id="datas" dataSource={this.records}
    fields={this.fields} placeholder="e.g. Item 1" enableVirtualization={true}
    allowFiltering={true} popupHeight="200px" >
      <Inject services={[VirtualScroll]} />
    </ComboBoxComponent>
  );
}
}

ReactDOM.render(<App />, document.getElementById('sample'));

```

Filtering with Virtualization

The ComboBox component supports Filtering with Virtualization. The ComboBox includes a built-in feature that enables data filtering when the [allowFiltering](#) option is enabled. In the context of Virtual Scrolling, the filtering process operates in response to the typed characters. Specifically, the DropDownList sends a request to the server, utilizing the full data source, to achieve filtering. Before initiating the request, an action event is triggered. Upon successful retrieval of data from the server, an action complete event is triggered. The initial data is loaded when the popup is opened. Whether the filter list has a selection or not, the popup closes.

The following sample shows the example for Filtering with Virtualization.

[Class-component]

INDEX.JSX

```

import { ComboBoxComponent, Inject, VirtualScroll } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  constructor(props) {
    super(props);
    this.records = Array.from({ length: 150 }, (_, i) => ({
      id: 'id' + (i + 1),
      text: `Item ${i + 1}`,
    }));
  }
  fields = { text: 'text', value: 'id' };
}

```

```
render() {
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="datas" dataSource={this.records}
        fields={this.fields} placeholder="e.g. Item 1" enableVirtualization={true}
        allowFiltering={true} popupHeight="200px" >
            <Inject services={[VirtualScroll]}/>
        </ComboBoxComponent>;
    )
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent, Inject, VirtualScroll } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // maps the appropriate column to fields property
    private fields: object = { text: 'text', value: 'id' };
    // define the array of string
    private records: { [key: string]: Object }[] = [];
    constructor(props) {
        super(props);
        this.records = Array.from({ length: 150 }, (_, i) => ({
            id: 'id' + (i + 1),
            text: `Item ${i + 1}`,
        }));
    }
    public render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="datas" dataSource={this.records}
            fields={this.fields} placeholder="e.g. Item 1" enableVirtualization={true}
            allowFiltering={true} popupHeight="200px" >
                <Inject services={[VirtualScroll]} />
            </ComboBoxComponent>
        );
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

Grouping in React Combo box component

The ComboBox supports wrapping nested elements into a group based on different categories. The category of each list item can be mapped through the `groupBy` field in the data table. The group header is displayed both as inline and fixed headers. The fixed group header content

is updated dynamically on scrolling the popup list with its category value.

In the following sample, the vegetables are grouped according to its category using `groupBy` field.

[Class-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    // define the data with category
    vegetableData = [
        { Vegetable: 'Cabbage', Category: 'Leafy and Salad', Id: 'item1' },
        { Vegetable: 'Spinach', Category: 'Leafy and Salad', Id: 'item2' },
        { Vegetable: 'Wheat grass', Category: 'Leafy and Salad', Id: 'item3' },
    ],
    { Vegetable: 'Yarrow', Category: 'Leafy and Salad', Id: 'item4' },
    { Vegetable: 'Pumpkins', Category: 'Leafy and Salad', Id: 'item5' },
    { Vegetable: 'Chickpea', Category: 'Beans', Id: 'item6' },
    { Vegetable: 'Green bean', Category: 'Beans', Id: 'item7' },
    { Vegetable: 'Horse gram', Category: 'Beans', Id: 'item8' },
    { Vegetable: 'Garlic', Category: 'Bulb and Stem', Id: 'item9' },
    { Vegetable: 'Nopal', Category: 'Bulb and Stem', Id: 'item10' },
    { Vegetable: 'Onion', Category: 'Bulb and Stem', Id: 'item11' }
    ];
    // map the groupBy field with Category column
    fields = { groupBy: 'Category', text: 'Vegetable', value: 'Id' };
    render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement" popupHeight='200px'
fields={this.fields} dataSource={this.vegetableData} placeholder="Select a
vegetable"/>);
        }
    }
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // define the data with category
    private vegetableData: { [key: string]: Object }[] = [
        { Vegetable: 'Cabbage', Category: 'Leafy and Salad', Id: 'item1' },
        { Vegetable: 'Spinach', Category: 'Leafy and Salad', Id: 'item2' },
        { Vegetable: 'Wheat grass', Category: 'Leafy and Salad', Id: 'item3' },
    ],
    { Vegetable: 'Yarrow', Category: 'Leafy and Salad', Id: 'item4' },
    { Vegetable: 'Pumpkins', Category: 'Leafy and Salad', Id: 'item5' },
    { Vegetable: 'Chickpea', Category: 'Beans', Id: 'item6' },
    { Vegetable: 'Green bean', Category: 'Beans', Id: 'item7' },
    { Vegetable: 'Horse gram', Category: 'Beans', Id: 'item8' },
    { Vegetable: 'Garlic', Category: 'Bulb and Stem', Id: 'item9' },
    { Vegetable: 'Nopal', Category: 'Bulb and Stem', Id: 'item10' },
    { Vegetable: 'Onion', Category: 'Bulb and Stem', Id: 'item11' }
    ];
    // map the groupBy field with Category column
    private fields: object = { groupBy: 'Category', text: 'Vegetable', value:
'Id' };
    public render() {
```

```

        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement" popupHeight='200px'
            fields={this.fields} dataSource={this.vegetableData} placeholder="Select a
            vegetable" />
        );
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

[Functional-component]

INDEX.JSX

```

import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // define the data with category
    const vegetableData = [
        { Vegetable: 'Cabbage', Category: 'Leafy and Salad', Id: 'item1' },
        { Vegetable: 'Spinach', Category: 'Leafy and Salad', Id: 'item2' },
        { Vegetable: 'Wheat grass', Category: 'Leafy and Salad', Id: 'item3' },
        { Vegetable: 'Yarrow', Category: 'Leafy and Salad', Id: 'item4' },
        { Vegetable: 'Pumpkins', Category: 'Leafy and Salad', Id: 'item5' },
        { Vegetable: 'Chickpea', Category: 'Beans', Id: 'item6' },
        { Vegetable: 'Green bean', Category: 'Beans', Id: 'item7' },
        { Vegetable: 'Horse gram', Category: 'Beans', Id: 'item8' },
        { Vegetable: 'Garlic', Category: 'Bulb and Stem', Id: 'item9' },
        { Vegetable: 'Nopal', Category: 'Bulb and Stem', Id: 'item10' },
        { Vegetable: 'Onion', Category: 'Bulb and Stem', Id: 'item11' }
    ];
    // map the groupBy field with Category column
    const fields = { groupBy: 'Category', text: 'Vegetable', value: 'Id' };
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" popupHeight='200px' fields={fields}
        dataSource={vegetableData} placeholder="Select a vegetable"/>
    );
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

INDEX.TSX

```

import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // define the data with category
    const vegetableData: { [key: string]: Object }[] = [
        { Vegetable: 'Cabbage', Category: 'Leafy and Salad', Id: 'item1' },
        { Vegetable: 'Spinach', Category: 'Leafy and Salad', Id: 'item2' },
        { Vegetable: 'Wheat grass', Category: 'Leafy and Salad', Id: 'item3' },
        { Vegetable: 'Yarrow', Category: 'Leafy and Salad', Id: 'item4' },
        { Vegetable: 'Pumpkins', Category: 'Leafy and Salad', Id: 'item5' },
    ];
}

```

```

    { Vegetable: 'Chickpea', Category: 'Beans', Id: 'item6' },
    { Vegetable: 'Green bean', Category: 'Beans', Id: 'item7' },
    { Vegetable: 'Horse gram', Category: 'Beans', Id: 'item8' },
    { Vegetable: 'Garlic', Category: 'Bulb and Stem', Id: 'item9' },
    { Vegetable: 'Nopal', Category: 'Bulb and Stem', Id: 'item10' },
    { Vegetable: 'Onion', Category: 'Bulb and Stem', Id: 'item11' }
  ];
  // map the groupBy field with Category column
  const fields: object = { groupBy: 'Category', text: 'Vegetable', value:
  'Id' };
  return (
    // specifies the tag for render the ComboBox component
    <ComboBoxComponent id="comboelement" popupHeight='200px'
fields={fields} dataSource={vegetableData} placeholder="Select a vegetable"
/>
  );
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

Customization

The grouping header is also provided with customization option. This allows custom designing using the `groupTemplate` property for both inline and fixed headers as referred here: [Group Template support to ComboBox](#).

See Also

- [Group Template support to ComboBox](#).

Filtering in React Combo box component

The ComboBox has built-in support to filter data items when `allowFiltering` is enabled. The filter operation starts as soon as you start typing characters in the component.

To display filtered items in the popup, filter the required data and return it to the ComboBox via `updateData` method by using the `filtering` event.

The following sample illustrates how to query the data source and pass the data to the ComboBox through the `updateData` method in `filtering` event.

[Class-component]

INDEX.JSX

```

// import DataManager related classes
import { Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  // define the filtering data
  searchData = [
    { Index: "s1", Country: "California" }, { Index: "s2", Country:
    "Florida" },
    { Index: "s3", Country: "Alaska" }, { Index: "s4", Country: "Georgia"
  }
  ];
}

```

```
// maps the appropriate column to fields property
fields = { text: "Country", value: "Index" };
// sort the resulted items
sortOrder = 'Ascending';
constructor(props) {
    super(props);
    this.onFiltering = this.onFiltering.bind(this);
}
// filtering event handler to filter a Country
onFiltering(args) {
    let query = new Query();
    // frame the query based on search string with filter type.
    query = (args.text !== "") ? query.where("Country", "startswith",
args.text, true) : query;
    // pass the filter data source, filter query to updateData method.
    args.updateData(this.searchData, query);
}
render() {
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" popupHeight="250px"
fields={this.fields} filtering={this.onFiltering} allowFiltering={true}
sortOrder={this.sortOrder} dataSource={this.searchData} placeholder="Select a
country"/>);
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
// import DataManager related classes
import { Query } from '@syncfusion/ej2-data';
import { SortOrder } from '@syncfusion/ej2-lists';
import { ComboBoxComponent, FilteringEventArgs } from '@syncfusion/ej2-react-
dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // define the filtering data
    private searchData: { [key: string]: Object }[] = [
        { Index: "s1", Country: "California" }, { Index: "s2", Country:
"Florida" },
        { Index: "s3", Country: "Alaska" }, { Index: "s4", Country: "Georgia"
}
    ];
    // maps the appropriate column to fields property
    private fields: object = { text: "Country", value: "Index" };
    // sort the resulted items
    private sortOrder: SortOrder = 'Ascending';
    constructor(props: any) {
        super(props);
        this.onFiltering = this.onFiltering.bind(this);
    }
    // filtering event handler to filter a Country
    public onFiltering(args: FilteringEventArgs) {
        let query = new Query();
```

```
// frame the query based on search string with filter type.
query = (args.text !== "") ? query.where("Country", "startswith",
args.text, true) : query;
// pass the filter data source, filter query to updateData method.
args.updateData(this.searchData, query);
}
public render() {
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" popupHeight="250px"
fields={this.fields} filtering={this.onFiltering} allowFiltering={true}
sortOrder={this.sortOrder} dataSource={this.searchData} placeholder="Select a
country" />
    );
}
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

[Functional-component]

INDEX.JSX

```
// import DataManager related classes
import { Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // define the filtering data
    const searchData = [
        { Index: "s1", Country: "California" }, { Index: "s2", Country:
"Florida" },
        { Index: "s3", Country: "Alaska" }, { Index: "s4", Country: "Georgia"
}
    ];
    // maps the appropriate column to fields property
    const fields = { text: "Country", value: "Index" };
    // sort the resulted items
    const sortOrder = 'Ascending';
    // filtering event handler to filter a Country
    function onFiltering(args) {
        let query = new Query();
        // frame the query based on search string with filter type.
        query = (args.text !== "") ? query.where("Country", "startswith",
args.text, true) : query;
        // pass the filter data source, filter query to updateData method.
        args.updateData(this.searchData, query);
    }
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" popupHeight="250px" fields={fields}
filtering={onFiltering} allowFiltering={true} sortOrder={sortOrder}
dataSource={searchData} placeholder="Select a country"/>
    );
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
// import DataManager related classes
import { Query } from '@syncfusion/ej2-data';
import { SortOrder } from '@syncfusion/ej2-lists';
import { ComboBoxComponent, FilteringEventArgs } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // define the filtering data
    const searchData: { [key: string]: Object }[] = [
        { Index: "s1", Country: "California" }, { Index: "s2", Country: "Florida" },
        { Index: "s3", Country: "Alaska" }, { Index: "s4", Country: "Georgia" }
    ];
    // maps the appropriate column to fields property
    const fields: object = { text: "Country", value: "Index" };
    // sort the resulted items
    const sortOrder: SortOrder = 'Ascending';
    // filtering event handler to filter a Country
    function onFiltering(args: FilteringEventArgs) {
        let query = new Query();
        // frame the query based on search string with filter type.
        query = (args.text !== "") ? query.where("Country", "startswith",
args.text, true) : query;
        // pass the filter data source, filter query to updateData method.
        args.updateData(this.searchData, query);
    }
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" popupHeight="250px"
fields={fields} filtering={onFiltering} allowFiltering={true}
sortOrder={sortOrder} dataSource={searchData} placeholder="Select a country"
/>
    );
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

Limit the minimum filter character

When filtering the list items, you can set the limit for character count to raise remote request and fetch filtered data on the ComboBox. This can be done by manual validation within the filter event handler.

In the following example, the remote request does not fetch the search data until the search key contains three characters.

[Class-component]

INDEX.JSX

```
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
```



```
// bind the DataManager instance to dataSource property
searchData = new DataManager({
  adaptor: new ODataV4Adaptor,
  crossDomain: true,
  url:
'https://services.odata.org/V4/Northwind/Northwind.svc/Customers'
});
// maps the appropriate column to fields property
fields = { text: 'ContactName', value: 'CustomerID' };
// bind the Query instance to query property
query = new Query().select(['ContactName', 'CustomerID']).take(6);
// sort the resulted items
sortOrder = 'Ascending';
constructor(props) {
  super(props);
  this.onFiltering = this.onFiltering.bind(this);
}
// filtering event handler to filter a customer
onFiltering(e) {
  // load overall data when search key empty.
  if (e.text === '') {
    e.updateData(this.searchData);
  }
  else {
    // restrict the remote request until search key contains 3
characters.
    if (e.text.length < 3) {
      return;
    }
    let query = new Query().select(['ContactName', 'CustomerID']);
    query = (e.text !== '') ? query.where('ContactName',
'startswith', e.text, true) : query;
    e.updateData(this.searchData, query);
  }
}
render() {
  return (
    // specifies the tag for render the ComboBox component
    <ComboBoxComponent id="comboelement" allowFiltering={true}
popupHeight="250px" filtering={this.onFiltering} query={this.query}
dataSource={this.searchData} fields={this.fields} placeholder="Select a
customer" sortOrder={this.sortOrder}/>);
}
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { SortOrder } from '@syncfusion/ej2-lists';
import { ComboBoxComponent, FilteringEventArgs } from '@syncfusion/ej2-react-
dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
  // bind the DataManager instance to dataSource property
```

```

private searchData: DataManager = new DataManager({
    adaptor: new ODataV4Adaptor,
    crossDomain: true,
    url:
'https://services.odata.org/V4/Northwind/Northwind.svc/Customers'
});
// maps the appropriate column to fields property
private fields: object = { text: 'ContactName', value: 'CustomerID' };
// bind the Query instance to query property
private query: Query = new Query().select(['ContactName',
'CustomerID']).take(6);
// sort the resulted items
private sortOrder: SortOrder = 'Ascending';
constructor(props: any) {
    super(props);
    this.onFiltering = this.onFiltering.bind(this);
}
// filtering event handler to filter a customer
public onFiltering(e: FilteringEventArgs) {
    // load overall data when search key empty.
    if (e.text === '') {
        e.updateData(this.searchData);
    } else {
        // restrict the remote request until search key contains 3
characters.
        if (e.text.length < 3) { return; }
        let query: Query = new Query().select(['ContactName',
'CustomerID']);
        query = (e.text !== '') ? query.where('ContactName',
'startswith', e.text, true) : query;
        e.updateData(this.searchData, query);
    }
}
public render() {
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" allowFiltering={true}
popupHeight="250px" filtering={this.onFiltering} query={this.query}
dataSource={this.searchData} fields={this.fields} placeholder="Select a
customer" sortOrder={this.sortOrder} />
    );
}
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

[Functional-component]

INDEX.JSX

```

import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // bind the DataManager instance to dataSource property
    const searchData = new DataManager({

```

```

        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url:
'https://services.odata.org/V4/Northwind/Northwind.svc/Customers'
    });
    // maps the appropriate column to fields property
    const fields = { text: 'ContactName', value: 'CustomerID' };
    // bind the Query instance to query property
    const query = new Query().select(['ContactName', 'CustomerID']).take(6);
    // sort the resulted items
    const sortOrder = 'Ascending';
    // filtering event handler to filter a customer
    function onFiltering(e) {
        // load overall data when search key empty.
        if (e.text === '') {
            e.updateData(this.searchData);
        }
        else {
            // restrict the remote request until search key contains 3
characters.
            if (e.text.length < 3) {
                return;
            }
            let query = new Query().select(['ContactName', 'CustomerID']);
            query = (e.text !== '') ? query.where('ContactName',
'startswith', e.text, true) : query;
            e.updateData(this.searchData, query);
        }
    }
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" allowFiltering={true}
popupHeight="250px" filtering={onFiltering} query={query}
dataSource={searchData} fields={fields} placeholder="Select a customer"
sortOrder={sortOrder}/>);
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

INDEX.TSX

```

import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { SortOrder } from '@syncfusion/ej2-lists';
import { ComboBoxComponent, FilteringEventArgs } from '@syncfusion/ej2-react-
dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // bind the DataManager instance to dataSource property
    const searchData: DataManager = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url:
'https://services.odata.org/V4/Northwind/Northwind.svc/Customers'
    });
    // maps the appropriate column to fields property
    const fields: object = { text: 'ContactName', value: 'CustomerID' };

```

```
// bind the Query instance to query property
const query: Query = new Query().select(['ContactName',
'CustomerID']).take(6);
// sort the resulted items
const sortOrder: SortOrder = 'Ascending';
// filtering event handler to filter a customer
function onFiltering(e: FilteringEventArgs) {
    // load overall data when search key empty.
    if (e.text === '') {
        e.updateData(this.searchData);
    } else {
        // restrict the remote request until search key contains 3
        characters.
        if (e.text.length < 3) { return; }
        let query: Query = new Query().select(['ContactName',
'CustomerID']);
        query = (e.text !== '') ? query.where('ContactName',
'startswith', e.text, true) : query;
        e.updateData(this.searchData, query);
    }
}
return (
    // specifies the tag for render the ComboBox component
    <ComboBoxComponent id="comboelement" allowFiltering={true}
popupHeight="250px" filtering={onFiltering} query={query}
dataSource={searchData} fields={fields} placeholder="Select a customer"
sortOrder={sortOrder} />
);
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

Change the filter type

While filtering, you can change the filter type to `contains`, `startsWith`, or `endsWith` for string type within the filter event handler.

In the following examples, data filtering is done with `endsWith` type.

[Class-component]

INDEX.JSX

```
// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    // bind the DataManager instance to dataSource property
    searchData = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url:
'https://services.odata.org/V4/Northwind/Northwind.svc/Customers'
    });
    // maps the appropriate column to fields property
    fields = { text: 'ContactName', value: 'CustomerID' };
}
```

```
// bind the Query instance to query property
query = new Query().select(['ContactName', 'CustomerID']).take(6);
// sort the resulted items
sortOrder = 'Ascending';
constructor(props) {
    super(props);
    this.onFiltering = this.onFiltering.bind(this);
}
// filtering event handler to filter a customer
onFiltering(e) {
    // load overall data when search key empty.
    if (e.text === '') {
        e.updateData(this.searchData);
    }
    else {
        let query = new Query().select(["ContactName", "CustomerID"]);
        query = (e.text !== '') ? query.where('ContactName', 'endswith',
e.text, true) : query;
        e.updateData(this.searchData, query);
    }
}
render() {
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" allowFiltering={true}
popupHeight="250px" filtering={this.onFiltering} query={this.query}
dataSource={this.searchData} fields={this.fields} placeholder="Select a
customer" sortOrder={this.sortOrder}/>);
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { SortOrder } from '@syncfusion/ej2-lists';
import { ComboBoxComponent, FilteringEventArgs } from '@syncfusion/ej2-react-
dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // bind the DataManager instance to dataSource property
    private searchData: DataManager = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url:
'https://services.odata.org/V4/Northwind/Northwind.svc/Customers'
    });
    // maps the appropriate column to fields property
    private fields: object = { text: 'ContactName', value: 'CustomerID' };
    // bind the Query instance to query property
    private query: Query = new Query().select(['ContactName',
'CustomerID']).take(6);
    // sort the resulted items
    private sortOrder: SortOrder = 'Ascending';
```

```

    constructor(props: any) {
        super(props);
        this.onFiltering = this.onFiltering.bind(this);
    }
    // filtering event handler to filter a customer
    public onFiltering(e: FilteringEventArgs) {
        // load overall data when search key empty.
        if (e.text === '') {
            e.updateData(this.searchData);
        } else {
            let query: Query = new Query().select(["ContactName",
"CustomerID"]);
            query = (e.text !== '') ? query.where('ContactName', 'endswith',
e.text, true) : query;
            e.updateData(this.searchData, query);
        }
    }
    public render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement" allowFiltering={true}
popupHeight="250px" filtering={this.onFiltering} query={this.query}
dataSource={this.searchData} fields={this.fields} placeholder="Select a
customer" sortOrder={this.sortOrder} />
        );
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

[Functional-component]

INDEX.JSX

```

// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // bind the DataManager instance to dataSource property
    const searchData = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url:
'https://services.odata.org/V4/Northwind/Northwind.svc/Customers'
    });
    // maps the appropriate column to fields property
    const fields = { text: 'ContactName', value: 'CustomerID' };
    // bind the Query instance to query property
    const query = new Query().select(['ContactName', 'CustomerID']).take(6);
    // sort the resulted items
    const sortOrder = 'Ascending';
    // filtering event handler to filter a customer
    function onFiltering(e) {
        // load overall data when search key empty.
        if (e.text === '') {

```

```

        e.updateData(this.searchData);
    }
    else {
        let query = new Query().select(["ContactName", "CustomerID"]);
        query = (e.text !== '') ? query.where('ContactName', 'endswith',
e.text, true) : query;
        e.updateData(this.searchData, query);
    }
}
return (
    // specifies the tag for render the ComboBox component
    <ComboBoxComponent id="comboelement" allowFiltering={true}
popupHeight="250px" filtering={onFiltering} query={query}
dataSource={searchData} fields={fields} placeholder="Select a customer"
sortOrder={sortOrder}/>);
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

INDEX.TSX

```

// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { SortOrder } from '@syncfusion/ej2-lists';
import { ComboBoxComponent, FilteringEventArgs } from '@syncfusion/ej2-react-
dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // bind the DataManager instance to dataSource property
    const searchData: DataManager = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url:
'https://services.odata.org/V4/Northwind/Northwind.svc/Customers'
    });
    // maps the appropriate column to fields property
    const fields: object = { text: 'ContactName', value: 'CustomerID' };
    // bind the Query instance to query property
    const query: Query = new Query().select(['ContactName',
'CustomerID']).take(6);
    // sort the resulted items
    const sortOrder: SortOrder = 'Ascending';
    // filtering event handler to filter a customer
    function onFiltering(e: FilteringEventArgs) {
        // load overall data when search key empty.
        if (e.text === '') {
            e.updateData(this.searchData);
        } else {
            let query: Query = new Query().select(["ContactName",
"CustomerID"]);
            query = (e.text !== '') ? query.where('ContactName', 'endswith',
e.text, true) : query;
            e.updateData(this.searchData, query);
        }
    }
    return (

```

```
// specifies the tag for render the ComboBox component
<ComboBoxComponent id="comboelement" allowFiltering={true}
popupHeight="250px" filtering={onFiltering} query={query}
dataSource={searchData} fields={fields} placeholder="Select a customer"
sortOrder={sortOrder} />
);
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

Case sensitive filtering

Data items can be filtered either with or without case sensitivity using the DataManager. This can be done by passing the fourth optional parameter of the **where** clause.

The following example shows how to perform case-sensitive filter.

[Class-component]

INDEX.JSX

```
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    // bind the DataManager instance to dataSource property
    searchData = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url:
'https://services.odata.org/V4/Northwind/Northwind.svc/Customers'
    });
    // maps the appropriate column to fields property
    fields = { text: 'ContactName', value: 'CustomerID' };
    // bind the Query instance to query property
    query = new Query().select(['ContactName', 'CustomerID']).take(6);
    // sort the resulted items
    sortOrder = 'Ascending';
    constructor(props) {
        super(props);
        this.onFiltering = this.onFiltering.bind(this);
    }
    // filtering event handler to filter a customer
    onFiltering(e) {
        // load overall data when search key empty.
        if (e.text === '') {
            e.updateData(this.searchData);
        }
        else {
            let query = new Query().select(["ContactName", "CustomerID"]);
            // enable the case sensitive filtering by passing false to 4th
parameter.
            query = (e.text !== '') ? query.where('ContactName', 'contains',
e.text, false) : query;
            e.updateData(this.searchData, query);
        }
    }
}
```



```
render() {
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" popupHeight="250px"
allowFiltering={true} filtering={this.onFiltering} query={this.query}
dataSource={this.searchData} fields={this.fields} placeholder="Select a
customer" sortOrder={this.sortOrder}/>);
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { SortOrder } from '@syncfusion/ej2-lists';
import { ComboBoxComponent, FilteringEventArgs } from '@syncfusion/ej2-react-
dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // bind the DataManager instance to dataSource property
    private searchData: DataManager = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url:
'https://services.odata.org/V4/Northwind/Northwind.svc/Customers'
    });
    // maps the appropriate column to fields property
    private fields: object = { text: 'ContactName', value: 'CustomerID' };
    // bind the Query instance to query property
    private query: Query = new Query().select(['ContactName',
'CustomerID']).take(6);
    // sort the resulted items
    private sortOrder: SortOrder = 'Ascending';
    constructor(props: any) {
        super(props);
        this.onFiltering = this.onFiltering.bind(this);
    }
    // filtering event handler to filter a customer
    public onFiltering(e: FilteringEventArgs) {
        // load overall data when search key empty.
        if (e.text === '') {
            e.updateData(this.searchData);
        } else {
            let query: Query = new Query().select(["ContactName",
"CustomerID"]);
            // enable the case sensitive filtering by passing false to 4th
parameter.
            query = (e.text !== '') ? query.where('ContactName', 'contains',
e.text, false) : query;
            e.updateData(this.searchData, query);
        }
    }
    public render() {
        return (
            // specifies the tag for render the ComboBox component
```

```

        <ComboBoxComponent id="comboelement" popupHeight="250px"
allowFiltering={true} filtering={this.onFiltering} query={this.query}
dataSource={this.searchData} fields={this.fields} placeholder="Select a
customer" sortOrder={this.sortOrder} />
    );
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

[Functional-component]

INDEX.JSX

```

import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // bind the DataManager instance to dataSource property
    const searchData = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url:
'https://services.odata.org/V4/Northwind/Northwind.svc/Customers'
    });
    // maps the appropriate column to fields property
    const fields = { text: 'ContactName', value: 'CustomerID' };
    // bind the Query instance to query property
    const query = new Query().select(['ContactName', 'CustomerID']).take(6);
    // sort the resulted items
    const sortOrder = 'Ascending';
    // filtering event handler to filter a customer
    function onFiltering(e) {
        // load overall data when search key empty.
        if (e.text === '') {
            e.updateData(this.searchData);
        }
        else {
            let query = new Query().select(["ContactName", "CustomerID"]);
            // enable the case sensitive filtering by passing false to 4th
parameter.
            query = (e.text !== '') ? query.where('ContactName', 'contains',
e.text, false) : query;
            e.updateData(this.searchData, query);
        }
    }
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" popupHeight="250px"
allowFiltering={true} filtering={onFiltering} query={query}
dataSource={searchData} fields={fields} placeholder="Select a customer"
sortOrder={sortOrder}/>);
}
ReactDOM.render(<App />, document.getElementById('sample'));

```

INDEX.TSX

```
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { SortOrder } from '@syncfusion/ej2-lists';
import { ComboBoxComponent, FilteringEventArgs } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';

function App() {
    // bind the DataManager instance to dataSource property
    const searchData: DataManager = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url:
'https://services.odata.org/V4/Northwind/Northwind.svc/Customers'
    });
    // maps the appropriate column to fields property
    const fields: object = { text: 'ContactName', value: 'CustomerID' };
    // bind the Query instance to query property
    const query: Query = new Query().select(['ContactName',
'CustomerID']).take(6);
    // sort the resulted items
    const sortOrder: SortOrder = 'Ascending';
    // filtering event handler to filter a customer
    function onFiltering(e: FilteringEventArgs) {
        // load overall data when search key empty.
        if (e.text === '') {
            e.updateData(this.searchData);
        } else {
            let query: Query = new Query().select(["ContactName",
'CustomerID']);
            // enable the case sensitive filtering by passing false to 4th
parameter.
            query = (e.text !== '') ? query.where('ContactName', 'contains',
e.text, false) : query;
            e.updateData(this.searchData, query);
        }
    }
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" popupHeight="250px"
allowFiltering={true} filtering={onFiltering} query={query}
dataSource={searchData} fields={fields} placeholder="Select a customer"
sortOrder={sortOrder} />
    );
}

ReactDOM.render(<App />, document.getElementById('sample'));
```

Diacritics Filtering

ComboBox supports diacritics filtering which will ignore the [diacritics](#) and makes it easier to filter the results in international characters lists when the [ignoreAccent](#) is enabled.

In the following sample,data with diacritics are bound as dataSource for ComboBox.

[Class-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    diacriticsData = [
        'Aeróbics',
        'Aeróbics en Agua',
        'Aerografía',
        'Aeromodelaje',
        'Águilas',
        'Ajedrez',
        'Ala Delta',
        'Álbumes de Música',
        'Alusivos',
        'Análisis de Escritura a Mano'
    ];
    render() {
        return (
            <ComboBoxComponent id="diacritics" ignoreAccent={true}
allowFiltering={true} dataSource={this.diacriticsData} placeholder="e.g:
aero"/>
        );
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    private diacriticsData: string[] = [
        'Aeróbics',
        'Aeróbics en Agua',
        'Aerografía',
        'Aeromodelaje',
        'Águilas',
        'Ajedrez',
        'Ala Delta',
        'Álbumes de Música',
        'Alusivos',
        'Análisis de Escritura a Mano'];
    public render() {
        return (
            <ComboBoxComponent id="diacritics" ignoreAccent={true}
allowFiltering={true} dataSource={this.diacriticsData} placeholder="e.g:
aero" />
        );
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

[Functional-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const diacriticsData = [
        'Aeróbics',
        'Aeróbics en Agua',
        'Aerografía',
        'Aeromodelaje',
        'Águilas',
        'Ajedrez',
        'Ala Delta',
        'Álbumes de Música',
        'Alusivos',
        'Análisis de Escritura a Mano'
    ];
    return (<ComboBoxComponent id="diacritics" ignoreAccent={true}
allowFiltering={true} dataSource={diacriticsData} placeholder="e.g: aero"/>);
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const diacriticsData: string[] = [
        'Aeróbics',
        'Aeróbics en Agua',
        'Aerografía',
        'Aeromodelaje',
        'Águilas',
        'Ajedrez',
        'Ala Delta',
        'Álbumes de Música',
        'Alusivos',
        'Análisis de Escritura a Mano'];
    return (
        <ComboBoxComponent id="diacritics" ignoreAccent={true}
allowFiltering={true} dataSource={diacriticsData} placeholder="e.g: aero" />
    );
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

See Also

- [How to achieve autofill while filtering](#)
- [How to group the data using header](#)

Localization in React Combo box component

The Localization library allows you to localize static text content of the [noRecordsTemplate](#) and [actionFailureTemplate](#) property according to the culture currently assigned to the ComboBox.

| Locale key | en-US (default) |

|-----|-----|

| noRecordsTemplate | No records found |

| actionFailureTemplate | The request failed |

Loading translations

To load translation object to your application, use load function of the **L10n** class.

In the following sample, French culture is set to the ComboBox and no data is loaded. Hence, the **noRecordsTemplate** property displays its text in French culture initially, and if the sample is run offline, the **actionFailureTemplate** property displays its text appropriately.

[Class-component]

INDEX.JSX

```
// import L10n class for load function
import { L10n } from '@syncfusion/ej2-base';
// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    // bind remotedata to showcase actionFailureTemplate in offline.
    customerData = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url:
'https://services.odata.org/V4/Northwind/Northwind.svc/Customers'
    });
    // maps the appropriate column to fields property
    fields = { text: 'ContactName', value: 'CustomerID' };
    // take 0 item to showcase noRecordsTemplate property.
    query = new Query().select(['ContactName', 'CustomerID']).take(0);
    // set locale culture to ComboBox
    componentWillMount() {
        L10n.load({
            'fr-BE': {
                'dropdowns': {
                    'actionFailureTemplate': "Modèle d'échec d'action",
                    'noRecordsTemplate': "Aucun enregistrement trouvé"
                }
            }
        });
    }
    render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement" fields={this.fields} locale="fr-
BE" query={this.query} dataSource={this.customerData}
placeholder="Sélectionnez un client"/>);
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
// import L10n class for load function
import { L10n } from '@syncfusion/ej2-base';
// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // bind remotedata to showcase actionFailureTemplate in offline.
    private customerData: DataManager = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url:
'https://services.odata.org/V4/Northwind/Northwind.svc/Customers'
    });
    // maps the appropriate column to fields property
    private fields: object = { text: 'ContactName', value: 'CustomerID' };
    // take 0 item to showcase noRecordsTemplate property.
    private query: Query = new Query().select(['ContactName',
'CustomerID']).take(0);
    // set locale culture to ComboBox
    public componentWillMount() {
        L10n.load({
            'fr-BE': {
                'dropdowns': {
                    'actionFailureTemplate': "Modèle d'échec d'action",
                    'noRecordsTemplate': "Aucun enregistrement trouvé"
                }
            }
        });
    }
    public render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement" fields={this.fields}
            locale="fr-BE" query={this.query} dataSource={this.customerData}
            placeholder="Sélectionnez un client" />
        );
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

[Functional-component]

INDEX.JSX

```
// import L10n class for load function
import { L10n } from '@syncfusion/ej2-base';
// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
```

```
export default class App extends React.Component {
    // bind remotedata to showcase actionFailureTemplate in offline.
    customerData = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url:
'https://services.odata.org/V4/Northwind/Northwind.svc/Customers'
    });
    // maps the appropriate column to fields property
    fields = { text: 'ContactName', value: 'CustomerID' };
    // take 0 item to showcase noRecordsTemplate property.
    query = new Query().select(['ContactName', 'CustomerID']).take(0);
    // set locale culture to ComboBox
    componentWillMount() {
        L10n.load({
            'fr-BE': {
                'dropdowns': {
                    'actionFailureTemplate': "Modèle d'échec d'action",
                    'noRecordsTemplate': "Aucun enregistrement trouvé"
                }
            }
        });
    }
    render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement" fields={this.fields} locale="fr-
BE" query={this.query} dataSource={this.customerData}
placeholder="Sélectionnez un client"/>);
        )
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
// import L10n class for load function
import { L10n } from '@syncfusion/ej2-base';
// import DataManager related classes
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // bind remotedata to showcase actionFailureTemplate in offline.
    private customerData: DataManager = new DataManager({
        adaptor: new ODataV4Adaptor,
        crossDomain: true,
        url:
'https://services.odata.org/V4/Northwind/Northwind.svc/Customers'
    });
    // maps the appropriate column to fields property
    private fields: object = { text: 'ContactName', value: 'CustomerID' };
    // take 0 item to showcase noRecordsTemplate property.
    private query: Query = new Query().select(['ContactName',
'CustomerID']).take(0);
    // set locale culture to ComboBox
```



```

public componentWillMount() {
    L10n.load({
        'fr-BE': {
            'dropdowns': {
                'actionFailureTemplate': "Modèle d'échec d'action",
                'noRecordsTemplate': "Aucun enregistrement trouvé"
            }
        }
    });
}

public render() {
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" fields={this.fields}
        locale="fr-BE" query={this.query} dataSource={this.customerData}
        placeholder="Sélectionnez un client" />
    );
}
}

ReactDOM.render(<App />, document.getElementById('sample'));

```

See Also

- [Accessibility](#)
- [How to bind the data to the combobox](#)

Style in React Combo box component

The following content provides the exact CSS structure that can be used to modify the control's appearance based on the user preference.

Customizing the appearance of wrapper element

Use the following CSS to customize the appearance of wrapper element.

```

`css
.e-ddl.e-input-group.e-control-wrapper .e-input {
font-size: 20px;
font-family: emoji;
color: #ab3243;
background: #32a5ab;
}
`

```

Customizing the dropdown icon's color

Use the following CSS to customize the dropdown icon's color.

```

`css
.e-ddl.e-input-group .e-input-group-icon,.e-ddl.e-input-group.e-control-wrapper .e-input-group-
icon:hover {

```

```
color: #bb233d;
font-size: 13px;
}
`
```

Customizing the focus color

Use the following CSS to customize the focusing color of input element.

```
`css
.e-ddl.e-input-group.e-control-wrapper.e-input-focus::before, .e-ddl.e-input-group.e-control-wrapper.e-
input-focus::after {
background: #c000ff;
}
`
```

Customizing the outline theme's focus color

Use the following CSS to customize the focusing color of outline theme.

```
`css
.e-outline.e-input-group.e-input-focus:hover:not(.e-success):not(.e-warning):not(.e-error):not(.e-
disabled):not(.e-float-icon-left),.e-outline.e-input-group.e-input-focus.e-control-wrapper:hover:not(.e-
success):not(.e-warning):not(.e-error):not(.e-disabled):not(.e-float-icon-left),.e-outline.e-input-group.e-
input-focus:not(.e-success):not(.e-warning):not(.e-error):not(.e-disabled),.e-outline.e-input-group.e-
control-wrapper.e-input-focus:not(.e-success):not(.e-warning):not(.e-error):not(.e-disabled) {
border-color: #b1bd15;
box-shadow: inset 1px 1px #b1bd15, inset -1px 0 #b1bd15, inset 0 -1px #b1bd15;
}
`
```

Customizing the disabled component's text color

Use the following CSS to customize the text color when the component is disabled.

```
`css
.e-input-group.e-control-wrapper .e-input[disabled] {
-webkit-text-fill-color: #0d9133;
}
`
```

Customizing the float label element's focusing color

Use the following CSS to customize the focusing color of float label element.

```
`css
.e-float-input.e-input-group:not(.e-float-icon-left) .e-float-line::before,.e-float-input.e-control-
wrapper.e-input-group:not(.e-float-icon-left) .e-float-line::before,.e-float-input.e-input-group:not(.e-
```

```
float-icon-left) .e-float-line::after,.e-float-input.e-control-wrapper.e-input-group:not(.e-float-icon-left)
.e-float-line::after {
background-color: #2319b8;
}

.e-ddl.e-input-group.e-control-wrapper.e-float-input.e-input-focus .e-float-text.e-label-top, .e-float-
input.e-control-wrapper:not(.e-error).e-input-focus input ~ label.e-float-text {
color: #2319b8;
}
、
```

Customizing the color of the placeholder text

Use the following CSS to customize the text color of placeholder.

```
`css
.e-ddl.e-input-group input.e-input::placeholder {
color: red;
}
、
```

Customizing the text selection color

Use the following CSS to customize the selection color of text and background.

```
`css
.e-ddl.e-input-group input.e-input::selection {
color: red;
background: yellow;
}
、
```

Customizing the background color of focus, hover, and active item's

Use the following CSS to customize the background color of focus, hover and active item's.

```
`css
.e-dropdownbase .e-list-item.e-item-focus, .e-dropdownbase .e-list-item.e-active, .e-dropdownbase .e-
list-item.e-active.e-hover, .e-dropdownbase .e-list-item.e-hover {
background-color: #1f9c99;
color: #2319b8;
}
、
```

Customizing the appearance of pop-up element

Use the following CSS to customize the appearance of popup element.

```
`css
.e-dropdownbase .e-list-item, .e-dropdownbase .e-list-item.e-item-focus {
background-color: #29c2b8;
color: #207cd9;
font-family: emoji;
min-height: 29px;
}
`
```

Adding mandatory asterisk to placeholder and float label

You can add a mandatory asterisk(*) to placeholder and float label using `.e-input-group.e-control-wrapper.e-float-input .e-float-text::after` class.

[Class-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  // define the array of data
  sportsData = ['Badminton', 'Cricket', 'Football', 'Golf', 'Tennis'];
  render() {
    return (
      // specifies the tag for render the ComboBox component
      <ComboBoxComponent id="comboelement" dataSource={this.sportsData}
placeholder="Select a game" floatLabelType="auto"/>);
  }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
  // define the array of data
  private sportsData: string[] = ['Badminton', 'Cricket', 'Football', 'Golf',
'Tennis'];
  public render() {
    return (
      // specifies the tag for render the ComboBox component
      <ComboBoxComponent id="comboelement" dataSource={this.sportsData}
placeholder="Select a game" floatLabelType= "auto"/>
    );
  }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

[Functional-component]

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // define the array of data
    const sportsData = ['Badminton', 'Cricket', 'Football', 'Golf', 'Tennis'];
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" dataSource={sportsData}
        placeholder="Select a game" floatLabelType="auto"/>);
    }
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // define the array of data
    const sportsData: string[] = ['Badminton', 'Cricket', 'Football', 'Golf', 'Tennis'];
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" dataSource={sportsData}
        placeholder="Select a game" floatLabelType= "auto"/>
    );
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

Accessibility in React Combo box component

The ComboBox component has been designed, keeping in mind the WAI-ARIA specifications, and applies the WAI-ARIA roles, states, and properties along with keyboard support. This component is characterized by complete keyboard interaction support and ARIA accessibility support that makes it easy for people who use assistive technologies (AT) or those who completely rely on keyboard navigation.

The ComboBox component followed the accessibility guidelines and standards, including [ADA](#), [Section 508](#), [WCAG 2.2](#) standards, and [WCAG roles](#) that are commonly used to evaluate accessibility.

The accessibility compliance for the ComboBox component is outlined below.

| Accessibility Criteria | Compatibility |

| -- | -- |

| [WCAG 2.2 Support](#) | |

| [Section 508 Support](#) | |

| [Screen Reader Support](#) | |

| [Right-To-Left Support](#) | |

| [Color Contrast](#) | |

| [Mobile Device Support](#) | |

| [Keyboard Navigation Support](#) | |

| [Accessibility Checker Validation](#) | |

| [Axe-core Accessibility Validation](#) | |

```
<style>
.post .post-content img {
display: inline-block;
margin: 0.5em 0;
}
</style>

<div> - All
features of the component meet the requirement.</div>

<div> - Some features of the component do not meet the requirement.</div>

<div> - The component does not meet the requirement.</div>
```

WAI-ARIA attributes

The ComboBox component followed the [WAI-ARIA](#) patterns to meet the accessibility. The following ARIA attributes are used in the ComboBox component:

| Properties | Functionalities |

| --- | --- |

| **aria-haspopup** | Indicates whether the ComboBox input element has a popup list or not. |

| **aria-expanded** | Indicates whether the popup list has expanded or not. |

| **aria-selected** | Indicates the selected option. |

| **aria-readonly** | Indicates the readonly state of the ComboBox element. |

- | aria-disabled | Indicates whether the ComboBox component is in a disabled state or not. |
- | aria-activedescendent | This attribute holds the ID of the active list item to focus its descendant child element. |
- | aria-owns | This attribute contains the ID of the popup list to indicate popup as a child element. |
- | aria-autocomplete | This attribute contains the 'both' to a list of options shows and the currently selected suggestion also shows inline. |

Keyboard interaction

You can use the following key shortcuts to access the ComboBox without interruptions.

| Keyboard shortcuts | Actions |

| --- | --- |

| Arrow Down | Selects the first item in the ComboBox when no item selected. Otherwise, selects the item next to the currently selected item. |

| Arrow Up | Selects the item previous to the currently selected one. |

| Page Down | Scrolls down to the next page and selects the first item when popup list opens. |

| Page Up | Scrolls up to the previous page and selects the first item when popup list opens. |

| Enter | Selects the focused item and popup list closes when it is in open state. |

| Tab | Focuses on the next TabIndex element on the page when the popup is closed. Otherwise, closes the popup list and remains the focus of the component. |

| Shift + tab | Focuses on the previous TabIndex element on the page when the popup is closed. Otherwise, closes the popup list and remains the focus of the component. |

| Alt + Down | Open the popup list |

| Alt + Up | Close the popup list |

| Esc(Escape) | Closes the popup list when it is in an open state and the currently selected item remains the same. |

| Home | Cursor moves to before of first character in input |

| End | Cursor moves to next of last character in input |

In the below sample, alt+t keys are used to focus the ComboBox component.

[Class-component]

INDEX.JSX

```
{% raw %}
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  // defined the array of data
  gameList = [
    { Id: 'Game2', Game: 'Badminton' },
    { Id: 'Game3', Game: 'Basketball' },
  ]
}
```

```

        { Id: 'Game4', Game: 'Cricket' },
        { Id: 'Game5', Game: 'Football' },
        { Id: 'Game6', Game: 'Golf' },
        { Id: 'Game7', Game: 'Hockey' },
        { Id: 'Game8', Game: 'Rugby' },
        { Id: 'Game9', Game: 'Snooker' },
        { Id: 'Game10', Game: 'Tennis' },
    ];
    // maps the appropriate column to fields property
    fields = { text: 'Game', value: 'Id' };
    // instance of ComboBox component
    comboBoxObj;
    componentDidMount() {
        const proxy = this;
        document.onkeyup = (e) => {
            if (e.altKey && e.keyCode === 84 /* t */) {
                // press alt+t to focus the control.
                proxy.comboBoxObj.inputElement.focus();
            }
        };
    }
    render() {
        return (
            // specifies the tag for render the ComboBox component
            <ComboBoxComponent id="comboelement" ref={(scope) => {
                this.comboBoxObj = scope; }} popupHeight='200px' fields={this.fields}
                dataSource={this.gameList} placeholder="Select a game"/>);
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // defined the array of data
    private gameList: { [key: string]: Object }[] = [
        { Id: 'Game2', Game: 'Badminton' },
        { Id: 'Game3', Game: 'Basketball' },
        { Id: 'Game4', Game: 'Cricket' },
        { Id: 'Game5', Game: 'Football' },
        { Id: 'Game6', Game: 'Golf' },
        { Id: 'Game7', Game: 'Hockey' },
        { Id: 'Game8', Game: 'Rugby' },
        { Id: 'Game9', Game: 'Snooker' },
        { Id: 'Game10', Game: 'Tennis' },
    ];
    // maps the appropriate column to fields property
    private fields: object = { text: 'Game', value: 'Id' };
    // instance of ComboBox component
    private comboBoxObj: ComboBoxComponent;
    public componentDidMount() {

```



```

const proxy = this;
document.onkeyup = (e) => {
  if (e.altKey && e.keyCode === 84 /* t */) {
    // press alt+t to focus the control.
    (proxy.comboBoxObj as any).inputElement.focus();
  }
};
}
public render() {
  return (
    // specifies the tag for render the ComboBox component
    <ComboBoxComponent id="comboelement" ref={(scope) => {
      (this.comboBoxObj as ComboBoxComponent | null) = scope; }}
      popupHeight='200px' fields={this.fields} dataSource={this.gameList}
      placeholder="Select a game" />
  );
}
}
ReactDOM.render(<App />, document.getElementById('sample'));
{% endraw %}

```

[Functional-component]

INDEX.JSX

```

{% raw %}
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  // defined the array of data
  const gameList = [
    { Id: 'Game2', Game: 'Badminton' },
    { Id: 'Game3', Game: 'Basketball' },
    { Id: 'Game4', Game: 'Cricket' },
    { Id: 'Game5', Game: 'Football' },
    { Id: 'Game6', Game: 'Golf' },
    { Id: 'Game7', Game: 'Hockey' },
    { Id: 'Game8', Game: 'Rugby' },
    { Id: 'Game9', Game: 'Snooker' },
    { Id: 'Game10', Game: 'Tennis' },
  ];
  // maps the appropriate column to fields property
  const fields = { text: 'Game', value: 'Id' };
  // instance of ComboBox component
  let comboBoxObj;
  React.useEffect(() => {
    const proxy = this;
    document.onkeyup = (e) => {
      if (e.altKey && e.keyCode === 84 /* t */) {
        // press alt+t to focus the control.
        proxy.comboBoxObj.inputElement.focus();
      }
    };
  }, []);
  return (

```

```
// specifies the tag for render the ComboBox component
<ComboBoxComponent id="comboelement" ref={ (scope) => { comboBoxObj =
scope; }} popupHeight='200px' fields={fields} dataSource={gameList}
placeholder="Select a game"/>;
}
ReactDOM.render(<App />, document.getElementById('sample'));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // defined the array of data
    const gameList: { [key: string]: Object }[] = [
        { Id: 'Game2', Game: 'Badminton' },
        { Id: 'Game3', Game: 'Basketball' },
        { Id: 'Game4', Game: 'Cricket' },
        { Id: 'Game5', Game: 'Football' },
        { Id: 'Game6', Game: 'Golf' },
        { Id: 'Game7', Game: 'Hockey' },
        { Id: 'Game8', Game: 'Rugby' },
        { Id: 'Game9', Game: 'Snooker' },
        { Id: 'Game10', Game: 'Tennis' },
    ];
    // maps the appropriate column to fields property
    const fields: object = { text: 'Game', value: 'Id' };
    // instance of ComboBox component
    let comboBoxObj: ComboBoxComponent;
    React.useEffect(() => {
        const proxy = this;
        document.onkeyup = (e) => {
            if (e.altKey && e.keyCode === 84 /* t */) {
                // press alt+t to focus the control.
                (proxy.comboBoxObj as any).inputElement.focus();
            }
        };
    }, []);
    return (
        // specifies the tag for render the ComboBox component
        <ComboBoxComponent id="comboelement" ref={ (scope) => { (comboBoxObj
as ComboBoxComponent | null) = scope; }} popupHeight='200px' fields={fields}
dataSource={gameList} placeholder="Select a game" />
    );
}
ReactDOM.render(<App />, document.getElementById('sample'));
{% endraw %}
```

Ensuring accessibility

The ComboBox component's accessibility levels are ensured through an [accessibility-checker](#) and [axe-core](#) software tools during automated testing.

The accessibility compliance of the ComboBox component is shown in the following sample. Open the [sample](#) in a new window to evaluate the accessibility of the ComboBox component with accessibility tools.

See also

- [Accessibility in Syncfusion React components](#)

How To

Autofill in React Combo box component

The ComboBox supports the **autofill** behaviour with the help of [autofill](#) property. Whenever you change the input value, the ComboBox will autocomplete your data by matching the typed character. Suppose, if no matches found then, ComboBox doesn't suggest any item.

The following examples, showcase that how to work autofill with ComboBox.

INDEX.JSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  // define the array of data
  sportsData = ['Badminton', 'Cricket', 'Football', 'Golf', 'Tennis'];
  render() {
    return (
      // specifies the tag for render the ComboBox component
      <ComboBoxComponent id="comboelement" placeholder="Select a game"
      autofill={true} dataSource={this.sportsData}/>);
  }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
  // define the array of data
  private sportsData: string[] = ['Badminton', 'Cricket', 'Football', 'Golf', 'Tennis'];
  public render() {
    return (
      // specifies the tag for render the ComboBox component
      <ComboBoxComponent id="comboelement" placeholder="Select a game"
      autofill={true} dataSource={this.sportsData} />
    );
  }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

Cascading in React Combo box component

The cascading ComboBox is a series of ComboBox, where the value of one ComboBox depends upon another's value. This can be configured by using the [change](#) event of the parent ComboBox. Within that change event handler, data has to be loaded to the child ComboBox based on the selected value of the parent ComboBox.

The following example, shows the cascade behavior of country, state, and city ComboBox. Here, the [dataBind](#) method is used to reflect the property changes immediately to the ComboBox.

INDEX.JSX

```
{% raw %}
import { Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  // country ComboBox instance
  countryObj;
  // state ComboBox instance
  stateObj;
  // city ComboBox instance
  cityObj;
  // define the country ComboBox data
  countryData = [
    { CountryName: 'Australia', CountryId: '2' },
    { CountryName: 'United States', CountryId: '1' }
  ];
  // define the state ComboBox data
  stateData = [
    { StateName: 'New York', CountryId: '1', StateId: '101' },
    { StateName: 'Virginia ', CountryId: '1', StateId: '102' },
    { StateName: 'Tasmania ', CountryId: '2', StateId: '105' }
  ];
  // define the city ComboBox data
  cityData = [
    { CityName: 'Albany', StateId: '101', CityId: 201 },
    { CityName: 'Beacon ', StateId: '101', CityId: 202 },
    { CityName: 'Emporia', StateId: '102', CityId: 206 },
    { CityName: 'Hampton ', StateId: '102', CityId: 205 },
    { CityName: 'Hobart', StateId: '105', CityId: 213 },
    { CityName: 'Launceston ', StateId: '105', CityId: 214 }
  ];
  // maps the country column to fields property
  countryField = { value: 'CountryId', text: 'CountryName' };
  // maps the state column to fields property
  stateField = { value: 'StateId', text: 'StateName' };
  // maps the city column to fields property
  cityField = { text: 'CityName', value: 'CityId' };
  constructor(props) {
    super(props);
    this.onCountryChange = this.onCountryChange.bind(this);
    this.onStateChange = this.onStateChange.bind(this);
  }
  onCountryChange() {
    // query the data source based on country ComboBox selected value
  }
}
```

```

        this.stateObj.query = new Query().where('CountryId', 'equal',
this.countryObj.value);
        // enable the state ComboBox
        this.stateObj.enabled = true;
        // clear the existing selection.
        this.stateObj.text = '';
        // bind the property changes to state ComboBox
        this.stateObj.dataBind();
        // clear the existing selection in city ComboBox
        this.cityObj.text = '';
        // disable the city ComboBox
        this.cityObj.enabled = false;
        // bind the property change to City ComboBox
        this.cityObj.dataBind();
    }
    onStateChange() {
        // query the data source based on state ComboBox selected value
        this.cityObj.query = new Query().where('StateId', 'equal',
this.stateObj.value);
        // enable the city ComboBox
        this.cityObj.enabled = true;
        // clear the existing selection
        this.cityObj.text = '';
        // bind the property change to city ComboBox
        this.cityObj.dataBind();
    }
    render() {
        return (<div>
            /* specifies the tag for render the country ComboBox
component */
            <ComboBoxComponent id="country-ddl" ref={(scope) => {
this.countryObj = scope; }} fields={this.countryField}
dataSource={this.countryData} placeholder='Select a country'
change={this.onCountryChange}/>
            <br />
            /* specifies the tag for render the state ComboBox
component */
            <ComboBoxComponent id="state-ddl" ref={(scope) => {
this.stateObj = scope; }} enabled={false} fields={this.stateField}
dataSource={this.stateData} placeholder='Select a state'
change={this.onStateChange}/>
            <br />
            /* specifies the tag for render the city ComboBox component
*/
            <ComboBoxComponent id="city-ddl" ref={(scope) => {
this.cityObj = scope; }} enabled={false} fields={this.cityField}
dataSource={this.cityData} placeholder='Select a city' />
            </div>);
    }
    ReactDOM.render(<App />, document.getElementById('sample'));
    {% endraw %}

```

INDEX.TSX

```
{% raw %}
```

```
import { Query } from '@syncfusion/ej2-data';
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // country ComboBox instance
    private countryObj: ComboBoxComponent;
    // state ComboBox instance
    private stateObj: ComboBoxComponent;
    // city ComboBox instance
    private cityObj: ComboBoxComponent;
    // define the country ComboBox data
    private countryData: { [key: string]: Object }[] = [
        { CountryName: 'Australia', CountryId: '2' },
        { CountryName: 'United States', CountryId: '1' }
    ];
    // define the state ComboBox data
    private stateData: { [key: string]: Object }[] = [
        { StateName: 'New York', CountryId: '1', StateId: '101' },
        { StateName: 'Virginia ', CountryId: '1', StateId: '102' },
        { StateName: 'Tasmania ', CountryId: '2', StateId: '105' }
    ];
    // define the city ComboBox data
    private cityData: { [key: string]: Object }[] = [
        { CityName: 'Albany', StateId: '101', CityId: 201 },
        { CityName: 'Beacon ', StateId: '101', CityId: 202 },
        { CityName: 'Emporia', StateId: '102', CityId: 206 },
        { CityName: 'Hampton ', StateId: '102', CityId: 205 },
        { CityName: 'Hobart', StateId: '105', CityId: 213 },
        { CityName: 'Launceston ', StateId: '105', CityId: 214 }
    ];
    // maps the country column to fields property
    private countryField: object = { value: 'CountryId', text: 'CountryName' };
    // maps the state column to fields property
    private stateField: object = { value: 'StateId', text: 'StateName' };
    // maps the city column to fields property
    private cityField: object = { text: 'CityName', value: 'CityId' };
    constructor(props: any) {
        super(props);
        this.onCountryChange=this.onCountryChange.bind(this);
        this.onStateChange=this.onStateChange.bind(this);
    }
    public onCountryChange() {
        // query the data source based on country ComboBox selected value
        this.stateObj.query = new Query().where('CountryId', 'equal',
this.countryObj.value);
        // enable the state ComboBox
        this.stateObj.enabled = true;
        // clear the existing selection.
        this.stateObj.text = '';
        // bind the property changes to state ComboBox
        this.stateObj.dataBind();
        // clear the existing selection in city ComboBox
        this.cityObj.text = '';
        // disable the city ComboBox
        this.cityObj.enabled = false;
    }
}
```

```

        // bind the property change to City ComboBox
        this.cityObj.dataBind();
    }
    public onStateChange() {
        // query the data source based on state ComboBox selected value
        this.cityObj.query = new Query().where('StateId', 'equal',
this.stateObj.value);
        // enable the city ComboBox
        this.cityObj.enabled = true;
        // clear the existing selection
        this.cityObj.text = '';
        // bind the property change to city ComboBox
        this.cityObj.dataBind();
    }
    public render() {
        return (
            <div>
                /* specifies the tag for render the country ComboBox
component */
                <ComboBoxComponent id="country-ddl" ref={(scope) => {
(this.countryObj as ComboBoxComponent | null) = scope; }}
fields={this.countryField} dataSource={this.countryData} placeholder='Select
a country' change={this.onCountryChange} />
                <br />
                /* specifies the tag for render the state ComboBox
component */
                <ComboBoxComponent id="state-ddl" ref={(scope) => {
(this.stateObj as ComboBoxComponent | null) = scope; }} enabled={false}
fields={this.stateField} dataSource={this.stateData} placeholder='Select a
state' change={this.onStateChange} />
                <br />
                /* specifies the tag for render the city ComboBox component
*/
                <ComboBoxComponent id="city-ddl" ref={(scope) => {
(this.cityObj as ComboBoxComponent | null) = scope; }} enabled={false}
fields={this.cityField} dataSource={this.cityData} placeholder='Select a
city' />
            </div>
        );
    }
}
ReactDOM.render(<App />, document.getElementById('sample'));
{% endraw %}

```

Icons support in React Combo box component

You can render **icons** to the list items by mapping the the **iconCss** fields. This **iconCss** field create a span in the list item with mapped class name to allow styling as per your need.

In the following sample, icon classes are mapped with **iconCss** field.

INDEX.JSX

```

import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {

```

```
// define the array of data
sortFormatData = [
  { Class: 'asc-sort', Type: 'Sort A to Z', Id: '1' },
  { Class: 'dsc-sort', Type: 'Sort Z to A ', Id: '2' },
  { Class: 'filter', Type: 'Filter', Id: '3' },
  { Class: 'clear', Type: 'Clear', Id: '4' }
];
// map the icon column to iconCSS field.
fields = { text: 'Type', iconCss: 'Class', value: 'Id' };
render() {
  return (
    // specifies the tag for render the ComboBox component
    <ComboBoxComponent id="comboelement" dataSource={this.sortFormatData}
    fields={this.fields} placeholder="Select a format"/>);
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

INDEX.TSX

```
import { ComboBoxComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
  // define the array of data
  private sortFormatData: { [key: string]: Object }[] = [
    { Class: 'asc-sort', Type: 'Sort A to Z', Id: '1' },
    { Class: 'dsc-sort', Type: 'Sort Z to A ', Id: '2' },
    { Class: 'filter', Type: 'Filter', Id: '3' },
    { Class: 'clear', Type: 'Clear', Id: '4' }
  ];
  // map the icon column to iconCSS field.
  private fields: object = { text: 'Type', iconCss: 'Class', value: 'Id' };
  public render() {
    return (
      // specifies the tag for render the ComboBox component
      <ComboBoxComponent id="comboelement"
      dataSource={this.sortFormatData} fields={this.fields} placeholder="Select a
      format" />
    );
  }
}
ReactDOM.render(<App />, document.getElementById('sample'));
```

Ej1 api migration in React Combo box component

This article describes the API migration process of ComboBox component from Essential JS 1 to Essential JS 2.

DataBinding

{% raw %}

<!-- markdownlint-disable MD010 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| **Default** | **Property:** *datasource*
<EJ.ComboBox dataSource={groups}></EJ.ComboBox> | **Property:** *dataSource*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} dataSource={this.cityData} /> |

| **Fields for mapping** | **Property:** *fields*
<EJ.ComboBox dataSource={groups} fields-value="parentId" fields-text="text"> </EJ.ComboBox> | **Property:** *fields*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} fields={this.field} /> |

| **Query** | **Property:** *query*
<EJ.ComboBox query={query}> </EJ.ComboBox> | **Property:** *query*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} query={this.query} /> |

| **Begin event** | **Event:** *actionBegin*
<EJ.ComboBox actionBegin="actionBegin"></EJ.ComboBox> | **Event:** *actionBegin*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} actionBegin={this.actionBegin.bind(this)} /> |

| **Complete event** | **Event:** *actionComplete*
<EJ.ComboBox actionComplete="actionComplete"></EJ.ComboBox> | **Event:** *actionComplete*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} actionComplete={this.actionComplete.bind(this)} /> |

| **Failure event** | **Event:** *actionFailure*
<EJ.ComboBox actionFailure="actionFailure"></EJ.ComboBox> | **Event:** *actionFailure*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} actionFailure={this.actionFailure.bind(this)} /> |

Filtering

<!-- markdownlint-disable MD010 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| **Default** | **Property:** *allowFiltering*
<EJ.ComboBox allowFiltering={true}></EJ.ComboBox> | **Property:** *allowFiltering*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} allowFiltering={true} /> |

| **No records template** | **Property:** *noRecordsTemplate*
<EJ.ComboBox noRecordsTemplate={template}></EJ.ComboBox> | **Property:** *noRecordsTemplate*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} noRecordsTemplate={this.noRecordsTemplate} /> |

| **Ignore casing and diacritics** | **Not Applicable** | **Property:** *ignoreAccent*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} ignoreAccent={true} /> |

| **Custom value addition** | **Property:** *allowCustom*
<EJ.ComboBox allowCustom={true}></EJ.ComboBox> | <https://ej2.syncfusion.com/react/demos/#/material/combo-box/custom-value> |

| **Search event** | **Event:** *filtering*
<EJ.ComboBox filtering="filtering"></EJ.ComboBox> | **Event:** *filtering*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} filtering={this.filteting.bind(this)} /> |

Template

<!-- markdownlint-disable MD010 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| **Default** | **Property:** *itemTemplate*
<EJ.ComboBox itemTemplate={itemTemplate}></EJ.ComboBox> | **Property:** *itemTemplate*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} itemTemplate={this.itemTemplate} />|

| **Group Template** | **Property:** *groupTemplate*
<EJ.ComboBox groupTemplate={groupTemplate}></EJ.ComboBox> | **Property:** *groupTemplate*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} groupTemplate={this.groupTemplate} />|

| **ValueTemplate** | **Not Applicable** | **Property:** *valueTemplate*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} valueTemplate={this.valueTemplate} /> |

| **Header Template** | **Property:** *headerTemplate*
<EJ.ComboBox headerTemplate={headerTemplate}></EJ.ComboBox> | **Property:** *headerTemplate*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} headerTemplate={this.headerTemplate} /> |

| **FooterTemplate** | **Property:** *footerTemplate*
<EJ.ComboBox footerTemplate={footerTemplate}></EJ.ComboBox> | **Property:** *footerTemplate*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} footerTemplate={this.footerTemplate} /> |

| **No records Template** | **Property:** *noRecordsTemplate*
<EJ.ComboBox noRecordsTemplate={noRecordsTemplate}></EJ.ComboBox> | **Property:** *noRecordsTemplate*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} noRecordsTemplate={this.noRecordsTemplate} /> |

| **Auto fill** | **Property:** *autoFill*
<EJ.ComboBox autoFill={true}></EJ.ComboBox> | **Property:** *autoFill*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} autoFill={true} />|

| **Action failure Template** | **Property:** *actionFailureTemplate*
<EJ.ComboBox actionFailureTemplate={actionFailureTemplate}></EJ.ComboBox> | **Property:** *actionFailureTemplate*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} actionFailureTemplate={this.actionFailureTemplate} />|

Applying CSS

<!-- markdownlint-disable MD010 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| **Default** | **Property:** *cssClass*
<EJ.ComboBox cssClass="cssClass"></EJ.ComboBox> | **Property:** *cssClass*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} cssClass={this.cssClass} /> |

| **width** | **Property:** *width*
<EJ.ComboBox width="300px"></EJ.ComboBox> | **Property:** *width*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} width={this.width} /> |

Grouping

<!-- markdownlint-disable MD010 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| **Default** | **Property:** *fields-groupBy*
<EJ.ComboBox dataSource={groups} fields-value="parentId" fields-groupBy="text"> </EJ.ComboBox> | **Property:** *fields*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} fields={this.field} /> |

Accessibility

<!-- markdownlint-disable MD010 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| **Globalization** | **Property:** *locale*
<EJ.ComboBox locale="fr-FE"></EJ.ComboBox> | **Property:** *locale*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} locale={this.locale} /> |

| **Rtl support** | **Property:** *enableRtl*
<EJ.ComboBox enableRtl={true}></EJ.ComboBox> | **Property:** *enableRtl*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} enableRtl={true} /> |

Placeholder

<!-- markdownlint-disable MD010 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| **Watermark text** | **Property:** *placeholder*
<EJ.ComboBox placeholder="select"></EJ.ComboBox> | **Property:** *placeholder*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} placeholder={this.placeholder} /> |

| **Floating of watermark text** | **Not applicable** | **Property:** *floatLabelType*
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} floatLabelType={this.floatLabelType} /> |

Miscellaneous

<!-- markdownlint-disable MD010 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| **Enable/disable** | **Property:** *enabled*
<EJ.ComboBox
enabled={true}></EJ.ComboBox> | **Property:** *enabled*
<ComboBoxComponent id="ddl"
ref={(scope) => { this.ddlObj = scope; }} enabled={true} /> |

| **Read only** | **Property:** *readOnly*
<EJ.ComboBox readOnly={true}></EJ.ComboBox> | **Property:**
readOnly
<ComboBoxComponent id="ddl" ref={(scope) => { this.ddlObj = scope; }}
readOnly={true} /> |

| **Addition of Html attributes** | **Property:** *htmlAttributes*
<EJ.ComboBox
htmlAttributes={htmlAttributes}></EJ.ComboBox> | **Property:**
htmlAttributes
<ComboBoxComponent id="ddl" ref={(scope) => { this.ddlObj = scope; }}
htmlAttributes={this.htmlAttributes} /> |

Sorting

<!-- markdownlint-disable MD010 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| **Order of sorting** | **Property:** *sortOrder*
<EJ.ComboBox
sortOrder={sortOrder}></EJ.ComboBox> | **Property:** *sortOrder*
<ComboBoxComponent
id="ddl" ref={(scope) => { this.ddlObj = scope; }} sortOrder={this.sortOrder} /> |

Selection

<!-- markdownlint-disable MD010 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| **Selecting particular index** | **Property:** *index*
<EJ.ComboBox index="1"></EJ.ComboBox> |
Property: *index*
<ComboBoxComponent id="ddl" ref={(scope) => { this.ddlObj = scope; }}
index={this.index} /> |

| **Selecting particular value** | **Property:** *value*
<EJ.ComboBox value="data"></EJ.ComboBox> |
Property: *value*
<ComboBoxComponent id="ddl" ref={(scope) => { this.ddlObj = scope; }}
value={this.value} /> |

| **Selecting particular text** | **Property:** *text*
<EJ.ComboBox text="data"></EJ.ComboBox> |
Property: *text*
<ComboBoxComponent id="ddl" ref={(scope) => { this.ddlObj = scope; }}
text={this.text} /> |

| **Getting data by using value** | **Method:** *getItemDataByValue*
<EJ.ComboBox
sortOrder={sortOrder}></EJ.ComboBox>

\$('#dropdown').ejComboBox('getItemDataByValue', 'data') | **Method:**
getDataByValue
<ComboBoxComponent id="ddl" ref={(scope) => { this.ddlObj = scope; }}
</>

 this.ddlObj.getDataByValue("data");

| **Select event** | **Event:** `select
<EJ.ComboBox select="select"></EJ.ComboBox>` | **Event:** `select
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }} select={this.select} />` |

Popup

<!-- markdownlint-disable MD010 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| **Popup height** | **Property:** `popupHeight
<EJ.ComboBox popupHeight="300px"></EJ.ComboBox>` | **Property:** `popupHeight
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }} popupHeight={this.popupHeight} />` |

| **Popup width** | **Property:** `popupWidth
<EJ.ComboBox popupWidth="300px"></EJ.ComboBox>` | **Property:** `popupWidth
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }} popupWidth={this.popupWidth} />` |

| **Popup showing manually** | **Method:** `showPopup
<EJ.ComboBox ></EJ.ComboBox>

$('#dropdown').ejComboBox("showPopup");` | **Method:** `showPopup
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }} />

this.ddlObj.showPopup();` |

| **Popup hiding manually** | **Method:** `hidePopup
<EJ.ComboBox ></EJ.ComboBox>

$('#dropdown').ejComboBox("hidePopup");` | **Method:** `hidePopup
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }} />

this.ddlObj.hidePopup();` |

| **Popup hide event** | **Event:** `close
<EJ.ComboBox close="close"></EJ.ComboBox>` | **Event:** `close
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }} close={this.close.bind(this)} />` |

| **Popup shown event** | **Event:** `open
<EJ.ComboBox open="open"></EJ.ComboBox>` | **Event:** `open
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }} open={this.open.bind(this)} />` |

Common

<!-- markdownlint-disable MD010 -->

| Behavior | API in Essential JS 1 | API in Essential JS 2 |

| --- | --- | --- |

| **Adding new item** | **Method :** `addItem
<EJ.ComboBox ></EJ.ComboBox>

$('#dropdown').ejComboBox("addItem", { text : "India"});` | **Method:** `addItem
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }} />

this.ddlObj.addItem({Id: 'id', Game: 'Golf'},2);` |

| **Focus out event** | **Not applicable** | **Event:** `blur
<ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }} blur={this.blur.bind(this)} />` |

```

| Focus in event | Event: focus<br/><EJ.ComboBox focus="focus"></EJ.ComboBox> | Event:
FocusIn<br/><ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}}
focusIn={this.focus.bind(this)} /> |

| Focus out | Method: focusOut<br/><EJ.ComboBox ></EJ.ComboBox> <br/>
<br/>${'#dropdown'}.ejComboBox("focusOut"); | Method: focusOut<br/><ComboBoxComponent
id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} /><br/><br/> this.ddlObj.focusOut(); |

| Focus in | Method: focusIn<br/><EJ.ComboBox ></EJ.ComboBox> <br/>
<br/>${'#dropdown'}.ejComboBox("focusIn"); | Method: focusIn<br/><ComboBoxComponent
id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} /><br/><br/> this.ddlObj.focusIn(); |

| Getting the data | Method : getItems<br/><EJ.ComboBox ></EJ.ComboBox> <br/>
<br/>${'#dropdown'}.ejComboBox("getItems"); | Method: getItems<br/><ComboBoxComponent
id="ddl" ref={{(scope) => { this.ddlObj = scope; }}} /><br/><br/> this.ddlObj.getItems(); |

| Create event | Event: create<br/><EJ.ComboBox close="close"></EJ.ComboBox> | Event:
created<br/><ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}}
created={this.create.bind(this)} /> |

| Change event | Event: change<br/><EJ.ComboBox close="close"></EJ.ComboBox> | Event:
change<br/><ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope; }}}
change={this.change.bind(this)} /> |

| Custom value event | Event: customValueSpecifier<br/><EJ.ComboBox
customValueSpecifier="customValueSpecifier"></EJ.ComboBox> | Event:
customValueSpecifier<br/><ComboBoxComponent id="ddl" ref={{(scope) => { this.ddlObj = scope;
}}} customValueSpecifier={this.customValueSpecifier.bind(this)} /> |

{% endraw %}

```

Context menu

Getting Started

This section explains how to create a simple ContextMenu, and configure its available functionalities in React

Dependencies

The following list of dependencies are required to use the ContextMenu component in your application.

```

`javascript
|-- @syncfusion/ej2-react-navigations
|-- @syncfusion/ej2-react-base
|-- @syncfusion/ej2-navigations
|-- @syncfusion/ej2-base
|-- @syncfusion/ej2-data
|-- @syncfusion/ej2-inputs
|-- @syncfusion/ej2-lists

```

```
|-- @syncfusion/ej2-popups
|-- @syncfusion/ej2-buttons
,
```

Setup your development environment

You can use [Create-react-app](#) to setup the applications.

To install `create-react-app` run the following command.

```
`bash
npm install -g create-react-app
,
```

Start a new project using create-react-app command as follows

```
<div class='tsx'>
`bash
create-react-app quickstart --scripts-version=react-scripts-ts
cd quickstart
,
```

```
</div>
<div class='jsx'>
`bash
create-react-app quickstart
cd quickstart
,
```

```
</div>
```

'react-scripts-ts' is used for creating React app with typescript.

Adding Syncfusion packages

All the available Essential JS 2 packages are published in [npmjs.com](https://www.npmjs.com) public registry.

To install `ContextMenu` component, use the following command

```
`bash
npm install @syncfusion/ej2-react-navigations --save
,
```

The above command installs [ContextMenu dependencies](#) which are required to render the component in the `React` environment.

Adding Style sheet to the Application

Add ContextMenu component's styles as given below in `App.css`.

```
`css
```

```
@import "../node_modules/@syncfusion/ej2-base/styles/material.css";
@import "../node_modules/@syncfusion/ej2-buttons/styles/material.css";
@import "../node_modules/@syncfusion/ej2-lists/styles/material.css";
@import "../node_modules/@syncfusion/ej2-popups/styles/material.css";
@import "../node_modules/@syncfusion/ej2-inputs/styles/material.css";
@import "../node_modules/@syncfusion/ej2-navigations/styles/material.css";
/ Context Menu target /
```

```
target {
border: 1px dashed;
height: 150px;
padding: 10px;
position: relative;
text-align: justify;
color: gray;
user-select: none;
}
,
```

Add ContextMenu to the project

Now, you can add `ContextMenu` component in the application. For getting started, add `ContextMenu` component in `src/App.tsx` file and the options contain `menuItems` and `target` in which `ContextMenu` will be opened. Using the following code snippet.

```
`ts
import { ContextMenuComponent, MenuItemModel } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import './App.css';
function App() {
let menuItems: MenuItemModel[] = [
{
text: 'Cut'
},
{
text: 'Copy'
},
{
```



```
text: 'Paste'
});
return (
  // specifies the tag to render the ContextMenu component
  <div>
    <div id="target">Right click / Touch hold to open the ContextMenu</div>
    <ContextMenuComponent target="#target" items={menuItems} />
  </div>
);
}
export default App;
`ts
import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import './App.css';
function App() {
  let menuItems = [
    {
      text: 'Cut'
    },
    {
      text: 'Copy'
    },
    {
      text: 'Paste'
    }
  ];
  return (
    // specifies the tag to render the ContextMenu component
    <div>
      <div id="target">Right click / Touch hold to open the ContextMenu</div>
      <ContextMenuComponent target="#target" items={menuItems}/>
    </div>
  );
}
```

```
</div>);
```

```
}
```

```
export default App;
```

```
,
```

[Run the application](#)

Run the application in the browser using the following command:

```
,
```

```
npm start
```

```
,
```

The following example shows a basic ContextMenu component.

INDEX.JSX

```
import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let menuItems = [
        {
            text: 'Cut'
        },
        {
            text: 'Copy'
        },
        {
            text: 'Paste'
        }
    ];
    return (
        <div className="container">
            <div id='target'>Right click / Touch hold to open the
            ContextMenu</div>
            <ContextMenuComponent id='contextmenu' target='#target'
            items={menuItems}></div>
        </div>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent, MenuItemModel } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let menuItems: MenuItemModel[] = [
        {
```

```

        text: 'Cut'
      },
      {
        text: 'Copy'
      },
      {
        text: 'Paste'
      }
    ]];
    return (
      <div className="container">
        <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
        <ContextMenuComponent id='contextmenu' target='#target'
          items={menuItems}/>
      </div>
    );
  }
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

See Also

- [ContextMenu with icons](#)
- [Multi-level nesting](#)

Icons and navigation in React Context menu component

Icons

The ContextMenu item have an icon / image in it to provide visual representation of the action. To place the icon on a menu item, set the [iconCss](#) property to e-icons with the required icon CSS. By default, the icon is positioned to the left side of the menu item. In the following sample, the icons for Cut, Copy and Paste menu items are added using `iconCss` property.

INDEX.JSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
  let menuItems = [
    {
      iconCss: 'e-cm-icons e-cut',
      text: 'Cut'
    },
    {
      iconCss: 'e-icons e-copy',
      text: 'Copy'
    },
    {
      iconCss: 'e-cm-icons e-paste',
      text: 'Paste'
    }
  ];
}

```

```

    return (<div className="container">
      <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
      <ContextMenuComponent id='contextmenu' target='#target'
items={menuItems}/>
    </div>);
  }
  export default App;
  ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent, MenuItemModel } from '@syncfusion/ej2-react-
navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
  let menuItems: MenuItemModel[] = [
    {
      iconCss: 'e-cm-icons e-cut',
      text: 'Cut'
    },
    {
      iconCss: 'e-icons e-copy',
      text: 'Copy'
    },
    {
      iconCss: 'e-cm-icons e-paste',
      text: 'Paste'
    }
  ];
  return (
    <div className="container">
      <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
      <ContextMenuComponent id='contextmenu' target='#target'
items={menuItems}/>
    </div>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

Navigation URL

Navigation URL in ContextMenu is used for navigating to other web page when menu item is clicked. This can be achieved by providing link to the menu item using the [url](#) property. In the following sample, Navigation URL for Flipkart, Amazon, and Snapdeal menu items are added using the [url](#) property.

INDEX.JSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';

```

```

enableRipple(true);
function App() {
  let menuItems = [
    {
      iconCss: 'e-cart-icon e-link',
      text: 'Flipkart',
      url: 'https://www.google.co.in/search?q=flipkart'
    },
    {
      iconCss: 'e-cart-icon e-link',
      text: 'Amazon',
      url: 'https://www.google.co.in/search?q=amazon'
    },
    {
      iconCss: 'e-cart-icon e-link',
      text: 'Snapdeal',
      url: 'https://www.google.co.in/search?q=snapdeal'
    }
  ];
  function itemBeforeEvent(args) {
    args.element.getElementsByTagName('a')[0].setAttribute('target',
    '_blank');
  }
  return (<div className="container">
    <div id="target">Right click / Touch hold to open the
    ContextMenu</div>
    <ContextMenuComponent id="contextmenu" target="#target"
    items={menuItems} beforeItemRender={itemBeforeEvent}/>
    </div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent, MenuEventArgs, MenuItemModel } from
 '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
  let menuItems: MenuItemModel[] = [
    {
      iconCss: 'e-cart-icon e-link',
      text: 'Flipkart',
      url: 'https://www.google.co.in/search?q=flipkart'
    },
    {
      iconCss: 'e-cart-icon e-link',
      text: 'Amazon',
      url: 'https://www.google.co.in/search?q=amazon'
    },
    {
      iconCss: 'e-cart-icon e-link',
      text: 'Snapdeal',

```

```

        url: 'https://www.google.co.in/search?q=snapdeal'
    }];
    function itemBeforeEvent(args: MenuEventArgs) {
        args.element.getElementsByTagName('a')[0].setAttribute('target',
        '_blank');
    }
    return (
        <div className="container">
            <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
            <ContextMenuComponent id='contextmenu' target='#target'
items={menuItems}
            beforeItemRender={itemBeforeEvent}/>
        </div>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

To open the links in new tab, set 'target' attribute with the value '_blank' in the [beforeItemRender](#) event.

See Also

- [How to change menu items dynamically](#)

Template in React Context menu component

The ContextMenu items can be customized using the [beforeItemRender](#) property. The item render event triggers while rendering each menu item. The event argument will be used to identify the menu item and customized it based on the requirement. In the following sample, the menu item is rendered with keycode for specified action in ContextMenu using the template. Here, the keycode is specified for Save as, View page source, and Inspect in the right side corner of the menu items by adding span element in the [beforeItemRender](#) event.

INDEX.JSX

```

import { createElement, enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let menuItems = [
        {
            text: 'Save as...'
        },
        {
            text: 'View page source'
        },
        {
            text: 'Inspect'
        }
    ];
    function itemBeforeEvent(args) {

```

```

        const shortCutSpan = createElement('span');
        const text = args.item.text;
        const shortCutText = text === 'Save as...' ? 'Ctrl + S' : (text ===
'View page source' ? 'Ctrl + U' : 'Ctrl + Shift + I');
        shortCutSpan.textContent = shortCutText;
        args.element.appendChild(shortCutSpan);
        shortCutSpan.setAttribute('class', 'shortcut');
    }
    return (<div class="container">
        <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
        <ContextMenuComponent id='contextmenu' target='#target'
items={menuItems} beforeItemRender={itemBeforeEvent}/>
    </div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { createElement, enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent, MenuEventArgs, MenuItemModel } from
'@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let menuItems: MenuItemModel[] = [
        {
            text: 'Save as...'
        },
        {
            text: 'View page source'
        },
        {
            text: 'Inspect'
        }
    ];
    function itemBeforeEvent(args: MenuEventArgs) {
        const shortCutSpan = createElement('span');
        const text = args.item.text;
        const shortCutText = text === 'Save as...' ? 'Ctrl + S' : (text ===
'View page source' ? 'Ctrl + U' : 'Ctrl + Shift + I');
        shortCutSpan.textContent = shortCutText;
        args.element.appendChild(shortCutSpan);
        shortCutSpan.setAttribute('class', 'shortcut');
    }
    return (
        <div class="container">
            <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
            <ContextMenuComponent id='contextmenu' target='#target'
items={menuItems} beforeItemRender={itemBeforeEvent}/>
        </div>
    );
}
export default App;

```

```
ReactDOM.render(<App />, document.getElementById('element'));
```

To create span element, `createElement` util function used from `ej2-base`.

Multilevel nesting

Multiple level nesting supports in ContextMenu. It can be achieved by mapping the [items](#) property inside the parent [menuItems](#). In the below sample, three level nesting of ContextMenu is provided.

INDEX.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
function App() {
    let menuItems = [
        {
            text: 'Show All Bookmarks'
        },
        {
            text: 'Bookmarks Toolbar',
            items: [
                {
                    text: 'Most Visited',
                    items: [
                        {
                            text: 'Google'
                        },
                        {
                            text: 'Gmail'
                        }
                    ]
                },
                {
                    text: 'Recently Added'
                }
            ]
        }
    ];
    return (<div class="container">
        <div id='target'>Right click / Touch hold to open the
        ContextMenu</div>
        <ContextMenuComponent id='contextmenu' target='#target'
        items={menuItems}> </ContextMenuComponent>
        </div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
```



```

import { ContextMenuComponent, MenuItemModel } from '@syncfusion/ej2-react-navigations';
import { enableRipple } from '@syncfusion/ej2-base';
enableRipple(true);
function App() {
  let menuItems: MenuItemModel[] = [
    {
      text: 'Show All Bookmarks'
    },
    {
      text: 'Bookmarks Toolbar',
      items: [
        {
          text: 'Most Visited',
          items: [
            {
              text: 'Google'
            },
            {
              text: 'Gmail'
            }
          ]
        },
        {
          text: 'Recently Added'
        }
      ]
    }
  ]];
  return (
    <div class="container">
      <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
      <ContextMenuComponent id='contextmenu' target='#target'
items={menuItems}> </ContextMenuComponent>
    </div>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

To open sub menu items only on click, [showItemOnClick](#) property should be set as **true**.

See Also

- [Populate menu items with data source](#)

Accessibility in React ContextMenu component

The ContextMenu component followed the accessibility guidelines and standards, including [ADA](#), [Section 508](#), [WCAG 2.2](#) standards, and [WCAG roles](#) that are commonly used to evaluate accessibility.

The accessibility compliance for the ContextMenu component is outlined below.

| Accessibility Criteria | Compatibility |

| -- | -- |

| [WCAG 2.2](#) Support | |

| [Section 508](#) Support | |

| Screen Reader Support | |

| Right-To-Left Support | |

| Color Contrast | |

| Mobile Device Support | |

| Keyboard Navigation Support | |

| [Accessibility Checker](#) Validation | |

| [Axe-core](#) Accessibility Validation | |

<style>

.post .post-content img {

display: inline-block;

margin: 0.5em 0;

}

</style>

<div> - All features of the component meet the requirement.</div>

<div> - Some features of the component do not meet the requirement.</div>

<div> - The component does not meet the requirement.</div>

WAI-ARIA attributes

The ContextMenu component followed the [WAI-ARIA](#) patterns to meet the accessibility. The following ARIA attributes are used in the ContextMenu component:

| Attributes | Purpose |

| --- | --- |

| `role` | Indicates ContextMenu component popup as `menu`, and the popup items as `menuitem`. |

| `aria-haspopup` | Indicates the availability and type of interactive popup element. |

| `aria-expanded` | Indicates whether the subtree can be expanded or collapsed, as well as indicates whether its current state is expanded or collapsed. |

| `aria-label` | Indicates the menu item text. |

Keyboard interaction

The ContextMenu component followed the [keyboard interaction](#) guideline, making it easy for people who use assistive technologies (AT) and those who completely rely on keyboard navigation. The following keyboard shortcuts are supported by the ContextMenu component.

| **Press** | **To do this** |

| --- | --- |

| **Esc** | Closes the opened sub menu. |

| **Enter** | Selects the focused item. |

| **Up** | Navigates up or to the previous menu item. |

| **Down** | Navigates down or to the next menu item. |

| **Left** | Close the current sub menu and navigates to the parent menu. |

| **Right** | Navigates and open the next sub menu. |

Ensuring accessibility

The ContextMenu component's accessibility levels are ensured through an [accessibility-checker](#) and [axe-core](#) software tools during automated testing.

The accessibility compliance of the ContextMenu component is shown in the following sample. Open the [sample](#) in a new window to evaluate the accessibility of the ContextMenu component with accessibility tools.

See also

- [Accessibility in Syncfusion React components](#)

Style and appearance in React Context menu component

To modify the ContextMenu appearance, you need to override the default CSS of ContextMenu component. Please find the list of CSS classes and its corresponding section in ContextMenu component. Also, you have an option to create your own custom theme for the controls using our [Theme Studio](#).

CSS Class | Purpose of Class

| `.e-contextmenu-wrapper` | To customize the context menu wrapper

| `.e-contextmenu-wrapper .e-menu-parent` | To customize the context menu items

| `.e-contextmenu-wrapper ul .e-menu-item.e-selected .e-caret::before` | To customize the context menu caret icon

| `.e-contextmenu-wrapper ul .e-menu-item .e-menu-icon::before` | To customize the icons of the context menu

How To

Data binding in React Context menu component

In the following example, menu items are populated from data source and mapped to [items](#) property.

INDEX.JSX

```
import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// @ts-ignore
import { data } from '../datasource.tsx';
function App() {
    function getMenuItems() {
        let record;
        const menuItems = [];
        for (const d of data) {
            record = d;
            if (record.parentId) {
                if (!menuItems[record.parentId - 1].items) {
                    menuItems[record.parentId - 1].items = [];
                }
                menuItems[record.parentId - 1].items.push({ text: record.text
});
            }
            else {
                menuItems.push({ text: record.text });
            }
        }
        return menuItems;
    }
    function itemBeforeEvent(args) {
        if (!args.item.text) {
            args.element.classList.add('e-separator');
        }
    }
    return (
        <div className="container">
            <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
            <ContextMenuComponent id='contextmenu' target='#target'
items={getMenuItems()} beforeItemRender={itemBeforeEvent}/>
        </div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import { ContextMenuComponent, MenuEventArgs, MenuItemModel } from
 '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// @ts-ignore
import { data, IRecord } from '../datasource.tsx';
function App() {
```

```

function getMenuItems() {
    let record: IRecord;
    const menuItems: MenuItemModel[] = [];
    for (const d of data) {
        record = d as IRecord;
        if (record.parentId) {
            if (!menuItems[record.parentId - 1].items) {
                menuItems[record.parentId - 1].items = [];
            }
            menuItems[record.parentId - 1].items.push({ text: record.text
});
        } else {
            menuItems.push({ text: record.text });
        }
    }
    return menuItems;
}
function itemBeforeEvent(args: MenuEventArgs) {
    if (!args.item.text) {
        args.element.classList.add('e-separator');
    }
}
return (
    <div className="container">
        <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
        <ContextMenuComponent id='contextmenu' target='#target'
            items={getMenuItems()} beforeItemRender = {itemBeforeEvent}
        />
    </div>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

While accessing Array we got the exception 'object is possibly undefined' due to 'strictNullChecks' option. So you can disable it in 'tsconfig.json' file.

Render with separator in React Context menu component

The Separators are horizontal lines that are used to separate the menu items. You cannot select the separators. You can enable separators to group the menu items using the [separator](#) property. Cut, Copy, and Paste menu items are grouped using `separator` property in the following sample.

INDEX.JSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let menuItems = [
        {
            text: 'Cut'
        },
    ],
    {

```

```

        text: 'Copy'
      },
      {
        text: 'Paste'
      },
      {
        separator: true
      },
      {
        text: 'Font'
      },
      {
        text: 'Paragraph'
      }
    ];
    return (<div className="container">
      <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
      <ContextMenuComponent id='contextmenu' target='#target'
items={menuItems}/>
    </div>);
  }
  export default App;
  ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent, MenuItemModel } from '@syncfusion/ej2-react-
navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
  let menuItems: MenuItemModel[] = [
    {
      text: 'Cut'
    },
    {
      text: 'Copy'
    },
    {
      text: 'Paste'
    },
    {
      separator: true
    },
    {
      text: 'Font'
    },
    {
      text: 'Paragraph'
    }
  ];
  return (
    <div className="container">

```

```

        <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
        <ContextMenuComponent id='contextmenu' target='#target'
items={menuItems}/>
    </div>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

The [separator](#) property should not be given along with

the other fields in the [MenuItem](#).

Open and close contextmenu in React Context menu component

ContextMenu can be opened and closed programmatically whenever required by using [open](#) and [close](#) methods.

In the following example, the ContextMenu is opened using the [open](#) method at the specified position using [top](#) and [left](#). Also, ContextMenu is closed using [close](#) method on ContextMenu item click or document click.

INDEX.JSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let cMenu;
    let menuItems = [
        {
            text: 'Cut'
        },
        {
            text: 'Copy'
        },
        {
            text: 'Paste'
        }
    ];
    function btnClick() {
        cMenu.open(40, 20);
    }
    return (<div className="container">
        <ContextMenuComponent id='contextmenu' ref={(scope) => cMenu =
scope} items={menuItems}/>
        <ButtonComponent onClick={btnClick}>Open
ContextMenu</ButtonComponent>
    </div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```
import { enableRipple } from '@syncfusion/ej2-base';
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
import { ContextMenuComponent, MenuItemModel } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let cMenu: ContextMenuComponent;
    let menuItems: MenuItemModel[] = [
        {
            text: 'Cut'
        },
        {
            text: 'Copy'
        },
        {
            text: 'Paste'
        }
    ];
    function btnClick(): void {
        cMenu.open(40, 20);
    }
    return (
        <div className="container">
            <ContextMenuComponent id='contextmenu' ref={(scope) => cMenu =
scope as ContextMenuComponent} items={menuItems} />
            <ButtonComponent onClick={btnClick}>Open
ContextMenu</ButtonComponent>
        </div>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));
```

Change menu items dynamically in React Context menu component

The items visible in the ContextMenu can be changed dynamically based on the context you open. To achieve this behavior, initialize ContextMenu with all items using [items](#) property and then based on the context you open hide/show required items using [hideItems](#)/[showItems](#) method in [beforeOpen](#) event.

In the following example, the datasource for Clipboard div is **Cut**, **Copy**, **Paste** and for the Editor div is **Add**, **Edit**, **Delete** is changed on [beforeOpen](#) event using [hideItems](#) and [showItems](#) method.

INDEX.JSX

```
import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let cmenuInstance;
    let menuItems = [
        {
```



```

        text: 'Cut'
    },
    {
        text: 'Copy'
    },
    {
        text: 'Paste'
    },
    {
        text: 'Add'
    },
    {
        text: 'Edit'
    },
    {
        text: 'Delete'
    }
];
function beforeOpen(args) {
    if (args.event.target.id === 'right') {
        cmenuInstance.hideItems(['Cut', 'Copy', 'Paste']);
        cmenuInstance.showItems(['Add', 'Edit', 'Delete']);
    }
    else if (args.event.target.id === 'left') {
        cmenuInstance.showItems(['Cut', 'Copy', 'Paste']);
        cmenuInstance.hideItems(['Add', 'Edit', 'Delete']);
    }
}
return (<div className="container">
    <div id="target">
        <div id='right' className='e-div'>Editor</div>
        <div id='left' className='e-div'>Clipboard</div>
    </div>
    <ContextMenuComponent id='contextmenu' target='#target'
ref={cmenu => cmenuInstance = cmenu} items={menuItems}
beforeOpen={beforeOpen}/>
    </div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { BeforeOpenCloseMenuEventArgs } from '@syncfusion/ej2-navigations';
import { ContextMenuComponent, MenuItemModel } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let cmenuInstance: ContextMenuComponent;
    let menuItems: MenuItemModel[] = [
        {
            text: 'Cut'
        },
    ],

```

```

    {
      text: 'Copy'
    },
    {
      text: 'Paste'
    },
    {
      text: 'Add'
    },
    {
      text: 'Edit'
    },
    {
      text: 'Delete'
    }
  ]];
function beforeOpen(args: BeforeOpenCloseMenuEventArgs) {
  if ((args.event.target as HTMLElement).id === 'right') {
    cmenuInstance.hideItems(['Cut', 'Copy', 'Paste']);
    cmenuInstance.showItems(['Add', 'Edit', 'Delete']);
  } else if ((args.event.target as HTMLElement).id === 'left') {
    cmenuInstance.showItems(['Cut', 'Copy', 'Paste']);
    cmenuInstance.hideItems(['Add', 'Edit', 'Delete']);
  }
}
return (
  <div className="container">
    <div id="target">
      <div id='right' className='e-div'>Editor</div>
      <div id='left' className='e-div'>Clipboard</div>
    </div>
    <ContextMenuComponent id='contextmenu' target='#target'
ref={cmenu => cmenuInstance = cmenu as ContextMenuComponent}
items={menuItems} beforeOpen={beforeOpen}/>
    </div>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

Scrollable contextmenu in React Context menu component

Scrollable ContextMenu can be achieved by restricting the height of the `ul` element.

In the following example, the `height` of the ContextMenu is set as `150px` and `overflow` property is set as `auto`.

INDEX.JSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
  let menuItems = [
    { text: 'ABS' },
    { text: 'ACOS' },

```

```

        { text: 'ACOSH' },
        { text: 'ACOT' },
        { text: 'ACOTH' },
        { text: 'AGGREGATE' },
        { text: 'COS' },
        { text: 'COSH' },
        { text: 'COT' },
        { text: 'COTH' }
    ];
    return (<div className="container">
        <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
        <ContextMenuComponent id='contextmenu' target='#target'
cssClass='e-custom' items={menuItems}/>
        </div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent, MenuItemModel } from '@syncfusion/ej2-react-
navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let menuItems: MenuItemModel[] = [
        { text: 'ABS' },
        { text: 'ACOS' },
        { text: 'ACOSH' },
        { text: 'ACOT' },
        { text: 'ACOTH' },
        { text: 'AGGREGATE' },
        { text: 'COS' },
        { text: 'COSH' },
        { text: 'COT' },
        { text: 'COTH' }
    ];
    return (
        <div className="container">
            <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
            <ContextMenuComponent id='contextmenu' target='#target'
cssClass='e-custom' items={menuItems} />
            </div>
        );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

Template in React Context menu component

Table in Sub ContextMenu

Menu items of the ContextMenu can be customized according to the requirement. The section explains about how to customize table template in sub menu item.

This can be achieved by appending table layout while `li` rendering by using `beforeItemRender` event.

INDEX.JSX

```
import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let menuItems = [
        {
            iconCss: 'e-cm-icons e-cut',
            text: 'Cut',
        },
        {
            iconCss: 'e-icons e-copy',
            text: 'Copy'
        },
        {
            iconCss: 'e-cm-icons e-paste',
            text: 'Paste'
        },
        {
            separator: true
        },
        {
            iconCss: 'e-icons e-link',
            text: 'Link'
        },
        {
            iconCss: 'e-icons e-table',
            items: [
                {
                    id: 'table'
                }
            ],
            text: 'Table'
        }
    ];
    function createHeader() {
        const header = document.createElement('h4');
        header.textContent = 'Insert Table';
        return header;
    }
    function createTable() {
        const table = document.createElement('table');
        for (let i = 0; i < 5; i++) {
            const row = document.createElement('tr');
            table.appendChild(row);
            for (let j = 0; j < 6; j++) {
                const col = document.createElement('td');
```

```

        row.appendChild(col);
        col.setAttribute('class', 'data');
    }
}
return table;
}
function itemBeforeEvent(args) {
    if (args.item.id === 'table') {
        args.element.classList.add('bg-transparent');
        args.element.appendChild(createHeader());
        args.element.appendChild(createTable());
    }
}
return (<div className="container">
    <div id='target'>Right click / Touch hold to open the
    ContextMenu</div>
    <ContextMenuComponent id='contextmenu' target='#target'
    items={menuItems} beforeItemRender={itemBeforeEvent}/>
    </div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent, MenuEventArgs, MenuItemModel } from
 '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let menuItems: MenuItemModel[] = [
        {
            iconCss: 'e-cm-icons e-cut',
            text: 'Cut',
        },
        {
            iconCss: 'e-icons e-copy',
            text: 'Copy'
        },
        {
            iconCss: 'e-cm-icons e-paste',
            text: 'Paste'
        },
        {
            separator: true
        },
        {
            iconCss: 'e-icons e-link',
            text: 'Link'
        },
        {
            iconCss: 'e-icons e-table',
            items: [
                {

```

```

        id: 'table'
      }
    ],
    text: 'Table'
  }
];
function createHeader() {
  const header = document.createElement('h4');
  header.textContent = 'Insert Table';
  return header;
}
function createTable() {
  const table = document.createElement('table');
  for (let i: number = 0; i < 5; i++) {
    const row = document.createElement('tr');
    table.appendChild(row);
    for (let j: number = 0; j < 6; j++) {
      const col = document.createElement('td');
      row.appendChild(col);
      col.setAttribute('class', 'data');
    }
  }
  return table;
}
function itemBeforeEvent(args: MenuEventArgs) {
  if (args.item.id === 'table') {
    args.element.classList.add('bg-transparent');
    args.element.appendChild(createHeader());
    args.element.appendChild(createTable());
  }
}
return (
  <div className="container">
    <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
    <ContextMenuComponent id='contextmenu' target='#target'
      items={menuItems} beforeItemRender={itemBeforeEvent}/>
  </div>
);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

UI Components in ContextMenu

UI components can also be placed inside the each `li` element of ContextMenu.

In the following example, CheckBox component is placed inside each `li` element and this can be achieved by creating CheckBox component in `beforeItemRender` event and appending it into the `li` element.

INDEX.JSX

```

import { closest, createElement, enableRipple } from '@syncfusion/ej2-base';
import { createCheckBox } from '@syncfusion/ej2-buttons';
import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';

```

```

enableRipple(true);
function App() {
  let menuItems = [
    { text: 'Option 1' },
    { text: 'Option 2' },
    { text: 'Option 3' }
  ];
  function itemBeforeEvent(args) {
    const check = createCheckBox(createElement, false, {
      checked: (args.item.text === 'Option 1' || args.item.text ===
'Option 2') ? true : false,
      label: args.item.text
    });
    args.element.innerHTML = '';
    args.element.appendChild(check);
  }
  function beforeClose(args) {
    if (closest(args.event.target, '.e-menu-item')) {
      args.cancel = true;
      const selectedElem = args.element.querySelectorAll('.e-
selected');
      for (const elem of selectedElem) {
        const ele = elem;
        ele.classList.remove('e-selected');
      }
      const checkbox = closest(args.event.target, '.e-checkbox-
wrapper');
      if (checkbox) {
        const frame = checkbox.querySelector('.e-frame');
        if (checkbox && frame.classList.contains('e-check')) {
          frame.classList.remove('e-check');
        }
        else if (checkbox) {
          frame.classList.add('e-check');
        }
      }
    }
  }
  return (<div className="container">
    <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
    <ContextMenuComponent id='contextmenu' target='#target'
items={menuItems} beforeItemRender={itemBeforeEvent}
beforeClose={beforeClose}/>
    </div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { closest, createElement, enableRipple} from '@syncfusion/ej2-base';
import { createCheckBox } from '@syncfusion/ej2-buttons';
import { BeforeOpenCloseMenuEventArgs, ContextMenuComponent, MenuEventArgs,
MenuItemModel } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';

```

```

import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
  let menuItems: MenuItemModel[] = [
    { text: 'Option 1' },
    { text: 'Option 2' },
    { text: 'Option 3' }
  ];
  function itemBeforeEvent(args: MenuEventArgs) {
    const check = createCheckBox(createElement, false, {
      checked: (args.item.text === 'Option 1' || args.item.text ===
'Option 2') ? true : false,
      label: args.item.text
    });
    args.element.innerHTML = '';
    args.element.appendChild(check);
  }
  function beforeClose(args: BeforeOpenCloseMenuEventArgs) {
    if (closest((args.event.target as HTMLElement), '.e-menu-item')) {
      args.cancel = true;
      const selectedElem = args.element.querySelectorAll('.e-
selected');
      for (const elem of selectedElem as any) {
        const ele = elem as HTMLElement;
        ele.classList.remove('e-selected');
      }
      const checkbox = closest(args.event.target as Element, '.e-
checkbox-wrapper') as HTMLElement;
      if (checkbox) {
        const frame = checkbox.querySelector('.e-frame');
        if (checkbox && frame.classList.contains('e-check')) {
          frame.classList.remove('e-check');
        } else if (checkbox) {
          frame.classList.add('e-check');
        }
      }
    }
  }
  return (
    <div className="container">
      <div id="target">Right click / Touch hold to open the
ContextMenu</div>
      <ContextMenuComponent id="contextmenu" target="#target"
items={menuItems} beforeItemRender={itemBeforeEvent}
beforeClose={beforeClose} />
    </div>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

Underline a character in the item text in React Context menu component

To underline a particular character in a text, it can be handled in `beforeItemRender` event by adding `<u>` tag in between the text and given as innerHTML in `li` rendering.

INDEX.JSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let menuItems = [
        {
            text: 'Cut'
        },
        {
            text: 'Copy'
        },
        {
            text: 'Paste'
        }
    ];
    function itemBeforeEvent(args) {
        if (args.item.text === 'Copy') {
            args.element.innerHTML = '<u>C</u>opy';
        }
    }
    return (<div className="container">
        <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
        <ContextMenuComponent id='contextmenu' target='#target'
items={menuItems} beforeItemRender={itemBeforeEvent}/>
        </div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent, MenuEventArgs, MenuItemModel } from
 '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let menuItems: MenuItemModel[] = [
        {
            text: 'Cut'
        },
        {
            text: 'Copy'
        },
        {
            text: 'Paste'
        }
    ];
    function itemBeforeEvent(args: MenuEventArgs) {
        if (args.item.text === 'Copy') {
            args.element.innerHTML = '<u>C</u>opy';
        }
    }
}

```

```

    }
  }
  return (
    <div className="container">
      <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
      <ContextMenuComponent id='contextmenu' target='#target'
items={menuItems} beforeItemRender={itemBeforeEvent}/>
    </div>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

Open a dialog on contextmenu item click in React Context menu component

This section explains about how to open a dialog on ContextMenu item click. This can be achieved by handling dialog open in `select` event of the ContextMenu.

In the following sample, Dialog will open while clicking `Save As...` item:

INDEX.JSX

```

import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import { DialogComponent } from '@syncfusion/ej2-react-popups';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  let dialogInstance;
  let position = { X: 100, Y: 100 };
  let visible = false;
  let buttons = [{
    buttonModel: {
      content: 'Submit',
      cssClass: 'e-flat',
      isPrimary: true
    },
    'click': () => {
      hide();
    }
  }];
  let menuItems = [
    {
      text: 'Back'
    },
    {
      text: 'Forward'
    },
    {
      text: 'Reload'
    },
    {
      separator: true
    },
    {
      text: 'Save As...'
    },
  ],

```

```

        {
            text: 'Print'
        },
        {
            text: 'Cast'
        }
    ];
    function hide() {
        dialogInstance.hide();
    }
    function select(args) {
        if (args.item.text === 'Save As...') {
            dialogInstance.show();
        }
    }
    return (<div className="container">
        <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
        <DialogComponent width='200px' height='110px' content='This file
can be saved as PDF' buttons={buttons} visible={visible} position={position}
ref={dialog => dialogInstance = dialog}/>
        <ContextMenuComponent id='contextmenu' target='#target'
items={menuItems} select={select}/>
    </div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { ContextMenuComponent, MenuEventArgs, MenuItemModel } from
'@syncfusion/ej2-react-navigations';
import { DialogComponent } from '@syncfusion/ej2-react-popups';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    let dialogInstance: DialogComponent;
    let position: any = { X: 100, Y: 100 };
    let visible: boolean = false;
    let buttons: any = [{
        buttonModel: {
            content: 'Submit',
            cssClass: 'e-flat',
            isPrimary: true
        },
    },
    {
        'click': () => {
            hide();
        }
    }
    ]];
    let menuItems: MenuItemModel[] = [
        {
            text: 'Back'
        },
        {
            text: 'Forward'
        },
    ],

```

```

        {
            text: 'Reload'
        },
        {
            separator: true
        },
        {
            text: 'Save As...'
        },
        {
            text: 'Print'
        },
        {
            text: 'Cast'
        }
    ]];
    function hide (): void {
        dialogInstance.hide();
    }
    function select(args: MenuEventArgs) {
        if (args.item.text === 'Save As...') {
            dialogInstance.show();
        }
    }
    return (
        <div className="container">
            <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
            <DialogComponent width='200px' height='110px' content='This file
can be saved as PDF' buttons={buttons} visible={visible} position={position}
ref={dialog => dialogInstance = dialog as DialogComponent} />
            <ContextMenuComponent id='contextmenu' target='#target'
items={menuItems} select={select}/>
        </div>
    );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

Change animation settings in React Context menu component

To change the animation of the ContextMenu, [animationSettings](#) property is used.

The supported effects for ContextMenu are,

Effect	Functionality
-----	-----
None	Specifies the sub menu transform with no animation effect.
SlideDown	Specifies the sub menu transform with slide down effect.
ZoomIn	Specifies the sub menu transform with zoom in effect.
FadeIn	Specifies the sub menu transform with fade in effect.

The following sample illustrates how to open ContextMenu with **FadeIn** effect with the **duration** of **800ms**.

INDEX.JSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let animation = {
        duration: 800,
        effect: 'FadeIn'
    };
    let menuItems = [
        {
            text: 'Show All Bookmarks'
        },
        {
            items: [
                {
                    items: [
                        {
                            text: 'Google'
                        },
                        {
                            text: 'Gmail'
                        }
                    ],
                    text: 'Most Visited'
                },
                {
                    text: 'Recently Added'
                }
            ],
            text: 'Bookmarks Toolbar'
        }
    ];
    return (
        <div className="container">
            <div id='target'>Right click / Touch hold to open the
            ContextMenu</div>
            <ContextMenuComponent id='contextmenu' target='#target'
            items={menuItems} animationSettings={animation}/>
            </div>);
    }
    export default App;
    ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent, MenuItemModel } from '@syncfusion/ej2-react-
navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let animation: any = {

```

```

    duration: 800,
    effect: 'FadeIn'
  };
  let menuItems: MenuItemModel[] = [
    {
      text: 'Show All Bookmarks'
    },
    {
      items: [
        {
          items: [
            {
              text: 'Google'
            },
            {
              text: 'Gmail'
            }
          ],
          text: 'Most Visited'
        },
        {
          text: 'Recently Added'
        }
      ],
      text: 'Bookmarks Toolbar'
    }
  ];
  return (
    <div className="container">
      <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
      <ContextMenuComponent id='contextmenu' target='#target'
        items={menuItems} animationSettings={animation}/>
    </div>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

Add or remove context menu items in React Context menu component

ContextMenu items can be added or removed using the [insertAfter](#), [insertBefore](#) and [removeItems](#) methods.

In the following example, the **Display Settings** menu items are added before the **Personalize** item, the **Sort By** menu items are added after the **Refresh**, and the **Paste** item is removed from context menu.

INDEX.JSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
  let cMenu;
  let menuItems = [
    {

```

```

        items: [
            {
                text: 'Large icons'
            },
            {
                text: 'Medium icons'
            },
            {
                text: 'Small icons'
            }
        ],
        text: 'View'
    },
    {
        text: 'Refresh'
    },
    {
        text: 'Paste'
    },
    {
        separator: true
    },
    {
        text: 'New'
    },
    {
        separator: true
    },
    {
        text: 'Personalize'
    }
];
function created() {
    cMenu.insertAfter([ { text: 'Sort By' } ], 'Refresh');
    cMenu.insertBefore([ { text: 'Display Settings' } ], 'Personalize');
    cMenu.removeItem(['Paste']);
}
;
return (<div className="container">
    <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
    <ContextMenuComponent id="cmenu" ref={ (scope) => cMenu = scope}
target='#target' items={menuItems} created={created}/>
    </div>);
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent, MenuItemModel } from '@syncfusion/ej2-react-
navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);

```

```

function App() {
  let cMenu: ContextMenuComponent;
  let menuItems: MenuItemModel[] = [
    {
      items: [
        {
          text: 'Large icons'
        },
        {
          text: 'Medium icons'
        },
        {
          text: 'Small icons'
        }
      ],
      text: 'View'
    },
    {
      text: 'Refresh'
    },
    {
      text: 'Paste'
    },
    {
      separator: true
    },
    {
      text: 'New'
    },
    {
      separator: true
    },
    {
      text: 'Personalize'
    }
  ]];
  function created(): void {
    cMenu.insertAfter([ {text: 'Sort By'} ], 'Refresh');
    cMenu.insertBefore([ {text: 'Display Settings'} ], 'Personalize');
    cMenu.removeItem(['Paste']);
  };
  return (
    <div className="container">
      <div id='target'>Right click / Touch hold to open the
ContextMenu</div>
      <ContextMenuComponent id="cmenu" ref={ (scope) => cMenu = scope
as ContextMenuComponent} target="#target" items={menuItems}
created={created}/>
    </div>
  );
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```


Enable or disable context menu items in React Context menu component

You can enable and disable the menu items using the [enableItems](#) method in ContextMenu. To enable menuItems, set the `enable` property in argument to `true` and vice-versa.

In the following example, the **Display Settings** in parent items and **Medium icons** in sub menu items are disabled.

INDEX.JSX

```
import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let cMenu;
    let menuItems = [
        {
            items: [
                { text: 'Large icons' },
                { text: 'Medium icons' },
                { text: 'Small icons' }
            ],
            text: 'View'
        },
        {
            text: 'Sort By'
        },
        {
            text: 'Refresh'
        },
        {
            separator: true
        },
        {
            text: 'New'
        },
        {
            separator: true
        },
        {
            text: 'Display Settings'
        },
        {
            text: 'Personalize'
        }
    ];
    return (
        <div className="container">
            <div id="target">Right click / Touch hold to open the
            ContextMenu</div>
            <ContextMenuComponent id="cmenu" ref={(scope) => cMenu = scope}
            target="#target" items={menuItems} created={created}
            beforeOpen={beforeOpen}/>
        </div>
    );
    function created() {
        cMenu.enableItems(['Display Settings'], false);
    }
}
```

```

;
function beforeOpen() {
    cMenu.enableItems(['Medium icons'], false);
}
;
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { enableRipple } from '@syncfusion/ej2-base';
import { ContextMenuComponent, MenuItemModel } from '@syncfusion/ej2-react-navigations';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
enableRipple(true);
function App() {
    let cMenu: ContextMenuComponent;
    let menuItemModel: MenuItemModel[] = [
        {
            items: [
                { text: 'Large icons' },
                { text: 'Medium icons' },
                { text: 'Small icons' }
            ],
            text: 'View'
        },
        {
            text: 'Sort By'
        },
        {
            text: 'Refresh'
        },
        {
            separator: true
        },
        {
            text: 'New'
        },
        {
            separator: true
        },
        {
            text: 'Display Settings'
        },
        {
            text: 'Personalize'
        }
    ];
    return (
        <div className="container">
            <div id="target">Right click / Touch hold to open the
ContextMenu</div>

```

```

        <ContextMenuComponent id="cmenu" ref={(scope) => cMenu = scope as
ContextMenuComponent} target='#target' items={menuItems} created={created}
beforeOpen={beforeOpen} />
    </div>
    );
    function created() {
        cMenu.enableItems(['Display Settings'], false);
    };
    function beforeOpen() {
        cMenu.enableItems(['Medium icons'], false);
    };
}
export default App;
ReactDOM.render(<App />, document.getElementById('element'));

```

To disable sub menu items, use the [beforeOpen](#) event.

Dashboard Layout

Getting Started

This section explains how to create a simple **Dashboard Layout** component and its basic usage.

Dependencies

The following list of dependencies is required to use the Dashboard Layout component in your application.

```

`js
|-- @syncfusion/ej2-react-layouts
|-- @syncfusion/ej2-react-base
|-- @syncfusion/ej2-base
|-- @syncfusion/ej2-layouts
,

```

Installation and configuration

You can use [create-react-app](#) to setup the applications.

To install `create-react-app` run the following command.

```

npm install -g create-react-app
,

```

To set-up a React application in TypeScript environment, run the following command.

```

`bash
npx create-react-app my-app --template typescript
cd my-app
npm start

```

`

To set-up a React application in JavaScript environment, run the following command.

```
`bash
npx create-react-app my-app
cd my-app
npm start
`
```

Adding Syncfusion packages

All the available Essential JS 2 packages are published in [npmjs.com](https://www.npmjs.com) public registry.

To install **Dashboard Layout** component, use the following command

```
`bash
npm install @syncfusion/ej2-react-layouts --save
`
```

Adding CSS Reference

To render the Dashboard Layout component, need to import Dashboard Layout and its dependent component's styles as given below in **src/App.css**.

```
`css
@import '../node_modules/@syncfusion/ej2-base/styles/material.css';
@import "../node_modules/@syncfusion/ej2-react-layouts/styles/material.css";
`
```

Note: If you want to refer the combined component styles, please make use of our [CRG](#) (Custom Resource Generator) in your application.

Add Dashboard Layout to the application

You can render the Dashboard Layout component in the following two ways.

- Defined the **panels** property as the attribute in the HTML element directly.
- Using the **panels** property directly.

Setting the `panels` property using HTML attributes

You can render the Dashboard Layout component by adding the panels property as the attribute to the HTML element. Add the HTML div element with panel definition for Dashboard Layout into your **App.tsx** file.

[src/App.tsx]

```
`ts
// import the DashboardLayout component
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
```

```
function App() {
let cellSpacing: number[] = [5, 5];
return (
<div>
<div className="control-section">
<DashboardLayoutComponent id='defaultLayout' cellSpacing={cellSpacing} allowResizing={true}
columns={5}>
<div id="one" className="e-panel" data-row="0" data-col="0" data-sizex="1" data-size="1">
<span id="close" className="e-template-icon e-clear-icon" />
<div className="e-panel-container">
<div className="text-align">0</div>
</div>
</div>
<div id="two" className="e-panel" data-row="1" data-col="0" data-sizex="1" data-size="2">
<span id="close" className="e-template-icon e-clear-icon" />
<div className="e-panel-container">
<div className="text-align">1</div>
</div>
</div>
<div id="three" className="e-panel" data-row="0" data-col="1" data-sizex="2" data-size="2">
<span id="close" className="e-template-icon e-clear-icon" />
<div className="e-panel-container">
<div className="text-align">2</div>
</div>
</div>
<div id="four" className="e-panel" data-row="2" data-col="1" data-sizex="1" data-size="1">
<span id="close" className="e-template-icon e-clear-icon" />
<div className="e-panel-container">
<div className="text-align">3</div>
</div>
</div>
<div id="five" className="e-panel" data-row="2" data-col="2" data-sizex="2" data-size="1">
<span id="close" className="e-template-icon e-clear-icon" />
```

```

<div className="e-panel-container">
  <div className="text-align">4</div>
</div>
</div>
<div id="six" className="e-panel" data-row="0" data-col="3" data-size="1" data-size="1">
  <span id="close" className="e-template-icon e-clear-icon" />
  <div className="e-panel-container">
    <div className="text-align">5</div>
  </div>
</div>
<div id="seven" className="e-panel" data-row="1" data-col="3" data-size="1" data-size="1">
  <span id="close" className="e-template-icon e-clear-icon" />
  <div className="e-panel-container">
    <div className="text-align">6</div>
  </div>
</div>
<div id="eight" className="e-panel" data-row="0" data-col="4" data-size="1" data-size="3">
  <span id="close" className="e-template-icon e-clear-icon" />
  <div className="e-panel-container">
    <div className="text-align">7</div>
  </div>
</div>
</DashboardLayoutComponent>
</div>
</div>
);
}
export default App;
`ts
// import the DashboardLayout component
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';

```

```
function App() {
  let cellSpacing = [5, 5];
  return (<div>
    <div className="control-section">
      <DashboardLayoutComponent id='defaultLayout' cellSpacing={cellSpacing} allowResizing={true}
        columns={5}>
        <div id="one" className="e-panel" data-row="0" data-col="0" data-size="1" data-size="1">
          <span id="close" className="e-template-icon e-clear-icon"/>
          <div className="e-panel-container">
            <div className="text-align">0</div>
          </div>
        </div>
        <div id="two" className="e-panel" data-row="1" data-col="0" data-size="1" data-size="2">
          <span id="close" className="e-template-icon e-clear-icon"/>
          <div className="e-panel-container">
            <div className="text-align">1</div>
          </div>
        </div>
        <div id="three" className="e-panel" data-row="0" data-col="1" data-size="2" data-size="2">
          <span id="close" className="e-template-icon e-clear-icon"/>
          <div className="e-panel-container">
            <div className="text-align">2</div>
          </div>
        </div>
        <div id="four" className="e-panel" data-row="2" data-col="1" data-size="1" data-size="1">
          <span id="close" className="e-template-icon e-clear-icon"/>
          <div className="e-panel-container">
            <div className="text-align">3</div>
          </div>
        </div>
        <div id="five" className="e-panel" data-row="2" data-col="2" data-size="2" data-size="1">
          <span id="close" className="e-template-icon e-clear-icon"/>
          <div className="e-panel-container">
```

```

<div className="text-align">4</div>
</div>
</div>
<div id="six" className="e-panel" data-row="0" data-col="3" data-size="1" data-size="1">
  <span id="close" className="e-template-icon e-clear-icon"/>
  <div className="e-panel-container">
    <div className="text-align">5</div>
  </div>
</div>
<div id="seven" className="e-panel" data-row="1" data-col="3" data-size="1" data-size="1">
  <span id="close" className="e-template-icon e-clear-icon"/>
  <div className="e-panel-container">
    <div className="text-align">6</div>
  </div>
</div>
<div id="eight" className="e-panel" data-row="0" data-col="4" data-size="1" data-size="3">
  <span id="close" className="e-template-icon e-clear-icon"/>
  <div className="e-panel-container">
    <div className="text-align">7</div>
  </div>
</div>
</DashboardLayoutComponent>
</div>
</div>);
}
export default App;
`

```

Run the application

Now, use the `npm start` command to run the application in the browser.

```

`bash
npm start
`

```

The following example shows a basic Dashboard Layout by adding the panels property directly into the HTML element.

APP.JSX

```

import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    const cellSpacing = [5, 5];
    return (<div>
        <div className="control-section">
            <DashboardLayoutComponent id='defaultLayout' cellSpacing={cellSpacing}
allowResizing={true} columns={5}>
                <div id="one" className="e-panel" data-row="0" data-col="0" data-
size="1" data-size="1">
                    <span id="close" className="e-template-icon e-clear-icon"/>
                    <div className="e-panel-container">
                        <div className="text-align">0</div>
                    </div>
                </div>
                <div id="two" className="e-panel" data-row="1" data-col="0" data-
size="1" data-size="2">
                    <span id="close" className="e-template-icon e-clear-icon"/>
                    <div className="e-panel-container">
                        <div className="text-align">1</div>
                    </div>
                </div>
                <div id="three" className="e-panel" data-row="0" data-col="1" data-
size="2" data-size="2">
                    <span id="close" className="e-template-icon e-clear-icon"/>
                    <div className="e-panel-container">
                        <div className="text-align">2</div>
                    </div>
                </div>
                <div id="four" className="e-panel" data-row="2" data-col="1" data-
size="1" data-size="1">
                    <span id="close" className="e-template-icon e-clear-icon"/>
                    <div className="e-panel-container">
                        <div className="text-align">3</div>
                    </div>
                </div>
                <div id="five" className="e-panel" data-row="2" data-col="2" data-
size="2" data-size="1">
                    <span id="close" className="e-template-icon e-clear-icon"/>
                    <div className="e-panel-container">
                        <div className="text-align">4</div>
                    </div>
                </div>
                <div id="six" className="e-panel" data-row="0" data-col="3" data-
size="1" data-size="1">
                    <span id="close" className="e-template-icon e-clear-icon"/>
                    <div className="e-panel-container">
                        <div className="text-align">5</div>
                    </div>
                </div>
                <div id="seven" className="e-panel" data-row="1" data-col="3" data-
size="1" data-size="1">
                    <span id="close" className="e-template-icon e-clear-icon"/>
                    <div className="e-panel-container">
                        <div className="text-align">6</div>
                    </div>
                </div>
            </DashboardLayoutComponent>
        </div>
    </div>
);
}

```

```

        </div>
      </div>
      <div id="eight" className="e-panel" data-row="0" data-col="4" data-
size="1" data-size="3">
        <span id="close" className="e-template-icon e-clear-icon"/>
        <div className="e-panel-container">
          <div className="text-align">7</div>
        </div>
      </div>
    </DashboardLayoutComponent>
  </div>
</div>;
}
export default App;

```

APP.TSX

```

import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
  const cellSpacing: number[] = [5, 5];
  return (
    <div>
      <div className="control-section">
        <DashboardLayoutComponent id='defaultLayout' cellSpacing={cellSpacing}
allowResizing={true} columns={5}>
          <div id="one" className="e-panel" data-row="0" data-col="0" data-
size="1" data-size="1">
            <span id="close" className="e-template-icon e-clear-icon" />
            <div className="e-panel-container">
              <div className="text-align">0</div>
            </div>
          </div>
          <div id="two" className="e-panel" data-row="1" data-col="0" data-
size="1" data-size="2">
            <span id="close" className="e-template-icon e-clear-icon" />
            <div className="e-panel-container">
              <div className="text-align">1</div>
            </div>
          </div>
          <div id="three" className="e-panel" data-row="0" data-col="1" data-
size="2" data-size="2">
            <span id="close" className="e-template-icon e-clear-icon" />
            <div className="e-panel-container">
              <div className="text-align">2</div>
            </div>
          </div>
          <div id="four" className="e-panel" data-row="2" data-col="1" data-
size="1" data-size="1">
            <span id="close" className="e-template-icon e-clear-icon" />
            <div className="e-panel-container">
              <div className="text-align">3</div>
            </div>
          </div>
          <div id="five" className="e-panel" data-row="2" data-col="2" data-
size="2" data-size="1">

```

```

        <span id="close" className="e-template-icon e-clear-icon" />
        <div className="e-panel-container">
            <div className="text-align">4</div>
        </div>
    </div>
    <div id="six" className="e-panel" data-row="0" data-col="3" data-
size="1" data-size="1">
        <span id="close" className="e-template-icon e-clear-icon" />
        <div className="e-panel-container">
            <div className="text-align">5</div>
        </div>
    </div>
    <div id="seven" className="e-panel" data-row="1" data-col="3" data-
size="1" data-size="1">
        <span id="close" className="e-template-icon e-clear-icon" />
        <div className="e-panel-container">
            <div className="text-align">6</div>
        </div>
    </div>
    <div id="eight" className="e-panel" data-row="0" data-col="4" data-
size="1" data-size="3">
        <span id="close" className="e-template-icon e-clear-icon" />
        <div className="e-panel-container">
            <div className="text-align">7</div>
        </div>
    </div>
</DashboardLayoutComponent>
</div>
</div>
);
}
export default App;

```

INDEX.JSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

INDEX.TSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

Setting the `panels` property directly

You can render the Dashboard Layout component by using the **panels** property directly.

[src/App.tsx]

```
`ts
```

```
// import the DashboardLayout component
```

```

import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
let cellSpacing: number[] = [5, 5];
let panels: object[] = [
{ "sizeX": 1, "sizeY": 1, "row": 0, "col": 0, content: '<div class="content">0</div>' },
{ "sizeX": 3, "sizeY": 2, "row": 0, "col": 1, content: '<div class="content">1</div>' },
{ "sizeX": 1, "sizeY": 3, "row": 0, "col": 4, content: '<div class="content">2</div>' },
{ "sizeX": 1, "sizeY": 1, "row": 1, "col": 0, content: '<div class="content">3</div>' },
{ "sizeX": 2, "sizeY": 1, "row": 2, "col": 0, content: '<div class="content">4</div>' },
{ "sizeX": 1, "sizeY": 1, "row": 2, "col": 2, content: '<div class="content">5</div>' },
{ "sizeX": 1, "sizeY": 1, "row": 2, "col": 3, content: '<div class="content">6</div>' }
];
return (
<div>
<div className="control-section">
<DashboardLayoutComponent id='defaultLayout' cellSpacing={cellSpacing} allowResizing={true}
panels={panels} columns={5} />
</div>
</div>
);
}
export default App;
`ts
// import the DashboardLayout component
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
let cellSpacing = [5, 5];
let panels = [
{ "sizeX": 1, "sizeY": 1, "row": 0, "col": 0, content: '<div class="content">0</div>' },
{ "sizeX": 3, "sizeY": 2, "row": 0, "col": 1, content: '<div class="content">1</div>' },

```

```

{ "sizeX": 1, "sizeY": 3, "row": 0, "col": 4, content: '<div class="content">2</div>' },
{ "sizeX": 1, "sizeY": 1, "row": 1, "col": 0, content: '<div class="content">3</div>' },
{ "sizeX": 2, "sizeY": 1, "row": 2, "col": 0, content: '<div class="content">4</div>' },
{ "sizeX": 1, "sizeY": 1, "row": 2, "col": 2, content: '<div class="content">5</div>' },
{ "sizeX": 1, "sizeY": 1, "row": 2, "col": 3, content: '<div class="content">6</div>' }
];

return (<div>
  <div className="control-section">
    <DashboardLayoutComponent id='defaultLayout' cellSpacing={cellSpacing} allowResizing={true}
    panels={panels} columns={5}/>
  </div>
</div>);
}

export default App;

```

The following example shows a basic Dashboard Layout by using the `panels` property.

APP.JSX

```

import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
  const cellSpacing = [5, 5];
  let panels = [
    { "sizeX": 1, "sizeY": 1, "row": 0, "col": 0, content: '<div class="content">0</div>' },
    { "sizeX": 3, "sizeY": 2, "row": 0, "col": 1, content: '<div class="content">1</div>' },
    { "sizeX": 1, "sizeY": 3, "row": 0, "col": 4, content: '<div class="content">2</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 1, "col": 0, content: '<div class="content">3</div>' },
    { "sizeX": 2, "sizeY": 1, "row": 2, "col": 0, content: '<div class="content">4</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 2, content: '<div class="content">5</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 3, content: '<div class="content">6</div>' }
  ];
  return (<div>
    <div className="control-section">
      <DashboardLayoutComponent id='defaultLayout'
      cellSpacing={cellSpacing} allowResizing={true} panels={panels} columns={5}/>
    </div>
  </div>);
}
export default App;

```

APP.TSX

```
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
  const cellSpacing: number[] = [5, 5];
  let panels: object[] = [
    { "sizeX": 1, "sizeY": 1, "row": 0, "col": 0, content: '<div class="content">0</div>' },
    { "sizeX": 3, "sizeY": 2, "row": 0, "col": 1, content: '<div class="content">1</div>' },
    { "sizeX": 1, "sizeY": 3, "row": 0, "col": 4, content: '<div class="content">2</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 1, "col": 0, content: '<div class="content">3</div>' },
    { "sizeX": 2, "sizeY": 1, "row": 2, "col": 0, content: '<div class="content">4</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 2, content: '<div class="content">5</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 3, content: '<div class="content">6</div>' }
  ];
  return (
    <div>
      <div className="control-section">
        <DashboardLayoutComponent id='defaultLayout'
          cellSpacing={cellSpacing} allowResizing={true} panels={panels} columns={5} />
      </div>
    </div>
  );
}
export default App;
```

INDEX.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));
```

INDEX.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));
```

You can refer to our [React Dashboard Layout](#) feature tour page for its groundbreaking feature representations. You can also explore our [React Dashboard Layout example](#) to know how to present and manipulate data.

Setting size of cells in React Dashboard layout component

The entire layout dimensions are assigned based on the height and width of the parent element. Hence, a responsive or static layout can be created by assigning a percentage or static dimension values to the parent element. The layout adapts to mobile resolutions by transforming the entire layout into a stacked orientation, so that, the panels will be displayed in a vertical column.

The **Dashboard Layout** is a grid structured component which can be split into subsections of equal size known as cells. The total number of cells in each row is defined by using the `columns` property of the component. The width of each cell will be auto calculated based on the total number of cells placed in a row and the height of a cell will be same as that of its width. However, the height of these cells can also be configured to any desired size using the `cellAspectRatio` property (cellwidth/cellheight ratio) which defines the cell width to height ratio.

The number of rows within the layout has no limits and can have any number of rows based on the panels count and position. Panels which acts as data containers will be placed or positioned over these cells.

Modifying cell size

In a dashboard, the data to be held by the panel in a cell may be of different size, hence different cell dimensions may be required in different scenarios. In this case, the size of these grid cells can be modified to the required size using the `columns` and `cellAspectRatio` properties.

The following sample demonstrates how to modify a cell size using the `columns` and `cellAspectRatio` properties. In the following sample, the width of the parent element is divided into 5 equal cells based on the `columns` property value resulting the width of each cell as 100 px. The height of these cells will be 50 px based on the `cellAspectRatio` value 100/50 (i.e., for every 100 px of width, 50 px will be the height of the cell).

APP.JSX

```
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
  const cellSpacing = [10, 10];
  let panels = [
    { "sizeX": 1, "sizeY": 1, "row": 0, "col": 0, content: '<div class="content">0</div>' },
    { "sizeX": 3, "sizeY": 2, "row": 0, "col": 1, content: '<div class="content">1</div>' },
    { "sizeX": 1, "sizeY": 3, "row": 0, "col": 4, content: '<div class="content">2</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 1, "col": 0, content: '<div class="content">3</div>' },
    { "sizeX": 2, "sizeY": 1, "row": 2, "col": 0, content: '<div class="content">4</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 2, content: '<div class="content">5</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 3, content: '<div class="content">6</div>' }
  ];
  return (
    <div>
      <div className="control-section">
```

```

        <DashboardLayoutComponent id='defaultLayout'
        cellSpacing={cellSpacing} cellAspectRatio={100 / 50} panels={panels}
        columns={5}/>
      </div>
    </div>);
  }
  export default App;

```

APP.TSX

```

import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
  const cellSpacing: number[] = [10, 10];
  let panels: object[] = [
    { "sizeX": 1, "sizeY": 1, "row": 0, "col": 0, content: '<div
class="content">0</div>' },
    { "sizeX": 3, "sizeY": 2, "row": 0, "col": 1, content: '<div
class="content">1</div>' },
    { "sizeX": 1, "sizeY": 3, "row": 0, "col": 4, content: '<div
class="content">2</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 1, "col": 0, content: '<div
class="content">3</div>' },
    { "sizeX": 2, "sizeY": 1, "row": 2, "col": 0, content: '<div
class="content">4</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 2, content: '<div
class="content">5</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 3, content: '<div
class="content">6</div>' }
  ];
  return (
    <div>
      <div className="control-section">
        <DashboardLayoutComponent id='defaultLayout'
        cellSpacing={cellSpacing} cellAspectRatio={100/50} panels={panels}
        columns={5} />
      </div>
    </div>
  );
}
export default App;

```

INDEX.JSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

INDEX.TSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```


Setting cell spacing

The spacing between each panel in a row and column can be defined using the `cellSpacing` property. Adding spacing between the panels will make the layout effective and provides a clear data representation.

The following sample demonstrates the usage of the `cellSpacing` property, which helps in a neat and clear representation of a data.

APP.JSX

```
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    const cellSpacing = [20, 20];
    let panels = [
        { "sizeX": 1, "sizeY": 1, "row": 0, "col": 0, content: '<div class="content">0</div>' },
        { "sizeX": 3, "sizeY": 2, "row": 0, "col": 1, content: '<div class="content">1</div>' },
        { "sizeX": 1, "sizeY": 3, "row": 0, "col": 4, content: '<div class="content">2</div>' },
        { "sizeX": 1, "sizeY": 1, "row": 1, "col": 0, content: '<div class="content">3</div>' },
        { "sizeX": 2, "sizeY": 1, "row": 2, "col": 0, content: '<div class="content">4</div>' },
        { "sizeX": 1, "sizeY": 1, "row": 2, "col": 2, content: '<div class="content">5</div>' },
        { "sizeX": 1, "sizeY": 1, "row": 2, "col": 3, content: '<div class="content">6</div>' }
    ];
    return (
        <div>
            <div className="control-section">
                <DashboardLayoutComponent id='defaultLayout'
                    cellSpacing={cellSpacing} panels={panels} columns={5}/>
            </div>
        </div>);
}
export default App;
```

APP.TSX

```
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    const cellSpacing: number[] = [20, 20];
    let panels: object[] = [
        { "sizeX": 1, "sizeY": 1, "row": 0, "col": 0, content: '<div class="content">0</div>' },
        { "sizeX": 3, "sizeY": 2, "row": 0, "col": 1, content: '<div class="content">1</div>' },
        { "sizeX": 1, "sizeY": 3, "row": 0, "col": 4, content: '<div class="content">2</div>' },
        { "sizeX": 1, "sizeY": 1, "row": 1, "col": 0, content: '<div class="content">3</div>' },
    ];
}
```

```

    { "sizeX": 2, "sizeY": 1, "row": 2, "col": 0, content: '<div
class="content">4</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 2, content: '<div
class="content">5</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 3, content: '<div
class="content">6</div>' }
  ];
  return (
    <div>
      <div className="control-section">
        <DashboardLayoutComponent id='defaultLayout'
cellSpacing={cellSpacing} panels={panels} columns={5} />
      </div>
    </div>
  );
}
export default App;

```

INDEX.JSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

INDEX.TSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

Graphical representation of layout

These cells combinedly forms a grid-structured layout which will be hidden initially. This grid structured layout can be made visible by enabling the `showGridLines` property, which clearly pictures the cells split-up within the layout. These gridlines will be helpful in panels sizing and placement within the layout during initial designing of a dashboard.

In the following sample, the gridlines indicate the cells split-up of the layout and the data containers placed over these cells are known as panels.

APP.JSX

```

import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
  const cellSpacing = [10, 10];
  let panels = [
    { "sizeX": 3, "sizeY": 2, "row": 0, "col": 1, content: '<div
class="content">1</div>' },
    { "sizeX": 1, "sizeY": 3, "row": 0, "col": 4, content: '<div
class="content">2</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 2, content: '<div
class="content">3</div>' },

```

```

    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 3, content: '<div
class="content">4</div>' }
  ];
  return (<div>
    <div className="control-section">
      <DashboardLayoutComponent id='defaultLayout'
cellSpacing={cellSpacing} showGridLines={true} panels={panels} columns={5}/>
    </div>
    </div>);
}
export default App;

```

APP.TSX

```

import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
  const cellSpacing: number[] = [10, 10];
  let panels: object[] = [
    { "sizeX": 3, "sizeY": 2, "row": 0, "col": 1, content: '<div
class="content">1</div>' },
    { "sizeX": 1, "sizeY": 3, "row": 0, "col": 4, content: '<div
class="content">2</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 2, content: '<div
class="content">3</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 3, content: '<div
class="content">4</div>' }
  ];
  return (
    <div>
      <div className="control-section">
        <DashboardLayoutComponent id='defaultLayout'
cellSpacing={cellSpacing} showGridLines={true} panels={panels} columns={5} />
      </div>
    </div>
  );
}
export default App;

```

INDEX.JSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

INDEX.TSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

Rendering component in right-to-left direction

It is possible to render the Dashboard Layout in right-to-left direction by setting the [enableRtl](#) API to true.

The following sample demonstrates Dashboard Layout in right-to-left direction.

APP.JSX

```
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    const cellSpacing = [10, 10];
    let enableRTL = true;
    let panels = [
        { 'id': 'Panel0', 'sizeX': 1, 'sizeY': 1, 'row': 0, 'col': 0, header:
        '<div>Panel 0</div>', content: '<div class="content">Content<div>' },
        { 'id': 'Panel1', 'sizeX': 3, 'sizeY': 2, 'row': 0, 'col': 1, header:
        '<div>Panel 1</div>', content: '<div class="content">Content<div>' },
        { 'id': 'Panel2', 'sizeX': 1, 'sizeY': 3, 'row': 0, 'col': 4, header:
        '<div>Panel 2</div>', content: '<div class="content">Content<div>' },
        { 'id': 'Panel3', 'sizeX': 1, 'sizeY': 1, 'row': 1, 'col': 0, header:
        '<div>Panel 3</div>', content: '<div class="content">Content<div>' },
        { 'id': 'Panel4', 'sizeX': 2, 'sizeY': 1, 'row': 2, 'col': 0, header:
        '<div>Panel 4</div>', content: '<div class="content">Content<div>' },
        { 'id': 'Panel5', 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 2, header:
        '<div>Panel 5</div>', content: '<div class="content">Content<div>' },
        { 'id': 'Panel6', 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 3, header:
        '<div>Panel 6</div>', content: '<div class="content">Content<div>' }
    ];
    return (<div>
        <div className="control-section">
            <DashboardLayoutComponent id='defaultLayout'
            cellSpacing={cellSpacing} enableRtl={enableRTL} panels={panels} columns={5}/>
        </div>
    </div>);
}
export default App;
```

APP.TSX

```
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App(){
    const cellSpacing: number[] = [10, 10];
    let enableRTL: boolean = true;
    let panels: object[] = [
        { 'id': 'Panel0', 'sizeX': 1, 'sizeY': 1, 'row': 0, 'col': 0,
        header: '<div>Panel 0</div>', content: '<div class="content">Content<div>' },
        { 'id': 'Panel1', 'sizeX': 3, 'sizeY': 2, 'row': 0, 'col': 1,
        header: '<div>Panel 1</div>', content: '<div class="content">Content<div>' },
        { 'id': 'Panel2', 'sizeX': 1, 'sizeY': 3, 'row': 0, 'col': 4,
        header: '<div>Panel 2</div>', content: '<div class="content">Content<div>' },
        { 'id': 'Panel3', 'sizeX': 1, 'sizeY': 1, 'row': 1, 'col': 0,
        header: '<div>Panel 3</div>', content: '<div class="content">Content<div>' },
        { 'id': 'Panel4', 'sizeX': 2, 'sizeY': 1, 'row': 2, 'col': 0,
        header: '<div>Panel 4</div>', content: '<div class="content">Content<div>' },
    ],
```

```

    {'id': 'Panel5', 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 2,
    header: '<div>Panel 5</div>', content: '<div class="content">Content</div>'},
    {'id': 'Panel6', 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 3,
    header: '<div>Panel 6</div>', content: '<div class="content">Content</div>'}
  ];
  return (
    <div>
      <div className="control-section">
        <DashboardLayoutComponent id='defaultLayout'
        cellSpacing={cellSpacing} enableRtl={enableRTL} panels={panels} columns={5}
        />
      </div>
    </div>
  );
}
export default App;

```

INDEX.JSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

INDEX.TSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

You can refer to our [React Dashboard Layout](#) feature tour page for its groundbreaking feature representations. You can also explore our [React Dashboard Layout example](#) to know how to present and manipulate data.

Panels**Position sizing of panels in React Dashboard layout component**

Panels are the basic building blocks of the dashboard layout component. They act as a container for the data to be visualized or presented. These panels can be positioned or resized for effective presentation of the data.

The following table represents all the available panel properties and the corresponding functionalities.

PanelObject	Description
---	---
id	Specifies the ID value of the panel.
row	Specifies the row value in which the panel to be placed.
col	Specifies the column value in which the panel to be placed.
sizeX	Specifies the width of the panel in cells count.

- | sizeY | Specifies the height of the panel in cells count. |
- | minSizeX | Specifies the minimum width of the panel in cells count. |
- | minSizeY | Specifies the minimum height of the panel in cells count. |
- | maxSizeX | Specifies the maximum width of the panel in cells count. |
- | maxSizeY | Specifies the maximum height of the panel in cells count. |
- | header | Specifies the header template of the panel. |
- | content | Specifies the content template of the panel. |
- | cssClass | Specifies the CSS class name that can be appended with each panel element. |

Positioning of panels

The panels within the layout can be easily positioned or ordered using the `row` and `col` properties of the panels. Positioning of panels will be beneficial to represent the data in any desired order.

The following sample demonstrates the positioning of panels within the dashboard layout using the row and column properties of the panels.

APP.JSX

```
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    const cellSpacing = [20, 20];
    let panels = [
        { "row": 0, "col": 0, content: '<div class="content">1</div>' },
        { "row": 0, "col": 1, content: '<div class="content">2</div>' },
        { "row": 0, "col": 2, content: '<div class="content">3</div>' },
        { "row": 1, "col": 0, content: '<div class="content">4</div>' },
        { "row": 1, "col": 1, content: '<div class="content">5</div>' },
        { "row": 1, "col": 2, content: '<div class="content">6</div>' }
    ];
    return (
        <div>
            <div className="container">
                <DashboardLayoutComponent id='defaultLayout'
                    cellSpacing={cellSpacing} panels={panels} columns={3}/>
            </div>
        </div>);
}
export default App;
```

APP.TSX

```
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App(){
    const cellSpacing: number[] = [20, 20];
    let panels: object[] = [
        { "row": 0, "col": 0, content: '<div class="content">1</div>' },
        { "row": 0, "col": 1, content: '<div class="content">2</div>' },
        { "row": 0, "col": 2, content: '<div class="content">3</div>' },
        { "row": 1, "col": 0, content: '<div class="content">4</div>' },
        { "row": 1, "col": 1, content: '<div class="content">5</div>' },
    ];
}
```

```

    { "row": 1, "col": 2, content: '<div class="content">6</div>' }
  ];
  return (
    <div>
      <div className="container">
        <DashboardLayoutComponent id='defaultLayout'
cellSpacing={cellSpacing} panels={panels} columns={3} />
      </div>
    </div>
  );
}
export default App;

```

INDEX.JSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

INDEX.TSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

Sizing of panels

A panel's size can be varied easily by defining the `sizeX` and `sizeY` properties. The `sizeX` property defines the width and the `sizeY` property defines height of a panel in cells count. These properties are helpful in designing a dashboard, where the content of each panel may vary in size.

The following sample demonstrates the sizing of panels within the dashboard layout using the `sizeX` and `sizeY` properties of the panels.

APP.JSX

```

import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
  const cellSpacing = [20, 20];
  let panels = [
    { "sizeX": 1, "sizeY": 1, "row": 0, "col": 0, content: '<div
class="content">0</div>' },
    { "sizeX": 3, "sizeY": 2, "row": 0, "col": 1, content: '<div
class="content">1</div>' },
    { "sizeX": 1, "sizeY": 3, "row": 0, "col": 4, content: '<div
class="content">2</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 1, "col": 0, content: '<div
class="content">3</div>' },
    { "sizeX": 2, "sizeY": 1, "row": 2, "col": 0, content: '<div
class="content">4</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 2, content: '<div
class="content">5</div>' },

```

```

        { "sizeX": 1, "sizeY": 1, "row": 2, "col": 3, content: '<div
class="content">6</div>' }
    ];
    return (<div>
        <div className="control-section">
            <DashboardLayoutComponent id='defaultLayout'
cellSpacing={cellSpacing} panels={panels} columns={5}/>
        </div>
    </div>);
}
export default App;

```

APP.TSX

```

import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    const cellSpacing: number[] = [20, 20];
    let panels: object[] = [
        { "sizeX": 1, "sizeY": 1, "row": 0, "col": 0, content: '<div
class="content">0</div>' },
        { "sizeX": 3, "sizeY": 2, "row": 0, "col": 1, content: '<div
class="content">1</div>' },
        { "sizeX": 1, "sizeY": 3, "row": 0, "col": 4, content: '<div
class="content">2</div>' },
        { "sizeX": 1, "sizeY": 1, "row": 1, "col": 0, content: '<div
class="content">3</div>' },
        { "sizeX": 2, "sizeY": 1, "row": 2, "col": 0, content: '<div
class="content">4</div>' },
        { "sizeX": 1, "sizeY": 1, "row": 2, "col": 2, content: '<div
class="content">5</div>' },
        { "sizeX": 1, "sizeY": 1, "row": 2, "col": 3, content: '<div
class="content">6</div>' }
    ];
    return (
        <div>
            <div className="control-section">
                <DashboardLayoutComponent id='defaultLayout'
cellSpacing={cellSpacing} panels={panels} columns={5} />
            </div>
        </div>
    );
}
export default App;

```

INDEX.JSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

INDEX.TSX

```

import * as React from 'react';

```



```
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));
```

You can refer to our [React Dashboard Layout](#) feature tour page for its groundbreaking feature representations. You can also explore our [React Dashboard Layout example](#) to know how to present and manipulate data.

Setting header of panels in React Dashboard layout component

The dashboard layout component is mostly used to represent the data used for monitoring or managing a process. These data or any HTML template can be placed as the content of a panel using the `content` property. Also, word or phrase that summarizes the panel's content can be added as the header on the top of each panel using the `header` property of the panel.

The following sample demonstrates how to add content for each panel using the header and content properties of the panels.

APP.JSX

```
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    const cellSpacing = [10, 10];
    let panels = [
        { 'id': 'Panel0', 'sizeX': 1, 'sizeY': 1, 'row': 0, 'col': 0, header:
        '<div>Panel 0</div>', content: '<div class="content">Panel Content</div>' },
        { 'id': 'Panel1', 'sizeX': 3, 'sizeY': 2, 'row': 0, 'col': 1, header:
        '<div>Panel 1</div>', content: '<div class="content">Panel Content</div>' },
        { 'id': 'Panel2', 'sizeX': 1, 'sizeY': 3, 'row': 0, 'col': 4, header:
        '<div>Panel 2</div>', content: '<div class="content">Panel Content</div>' },
        { 'id': 'Panel3', 'sizeX': 1, 'sizeY': 1, 'row': 1, 'col': 0, header:
        '<div>Panel 3</div>', content: '<div class="content">Panel Content</div>' },
        { 'id': 'Panel4', 'sizeX': 2, 'sizeY': 1, 'row': 2, 'col': 0, header:
        '<div>Panel 4</div>', content: '<div class="content">Panel Content</div>' },
        { 'id': 'Panel5', 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 2, header:
        '<div>Panel 5</div>', content: '<div class="content">Panel Content</div>' },
        { 'id': 'Panel6', 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 3, header:
        '<div>Panel 6</div>', content: '<div class="content">Panel Content</div>' }
    ];
    return (
        <div>
            <div className="control-section">
                <DashboardLayoutComponent id='defaultLayout'
                cellSpacing={cellSpacing} panels={panels} columns={5}/>
            </div>
        </div>);
}
export default App;
```

APP.TSX

```
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    const cellSpacing: number[] = [10, 10];
    let panels: object[] = [
```

```

    {'id': 'Panel0', 'sizeX': 1, 'sizeY': 1, 'row': 0, 'col': 0,
    header: '<div>Panel 0</div>', content: '<div class="content">Panel
    Content<div>'},
    {'id': 'Panel1', 'sizeX': 3, 'sizeY': 2, 'row': 0, 'col': 1,
    header: '<div>Panel 1</div>', content: '<div class="content">Panel
    Content<div>'},
    {'id': 'Panel2', 'sizeX': 1, 'sizeY': 3, 'row': 0, 'col': 4,
    header: '<div>Panel 2</div>', content: '<div class="content">Panel
    Content<div>'},
    {'id': 'Panel3', 'sizeX': 1, 'sizeY': 1, 'row': 1, 'col': 0,
    header: '<div>Panel 3</div>', content: '<div class="content">Panel
    Content<div>'},
    {'id': 'Panel4', 'sizeX': 2, 'sizeY': 1, 'row': 2, 'col': 0,
    header: '<div>Panel 4</div>', content: '<div class="content">Panel
    Content<div>'},
    {'id': 'Panel5', 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 2,
    header: '<div>Panel 5</div>', content: '<div class="content">Panel
    Content<div>'},
    {'id': 'Panel6', 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 3,
    header: '<div>Panel 6</div>', content: '<div class="content">Panel
    Content<div>'}
  ];
  return (
    <div>
      <div className="control-section">
        <DashboardLayoutComponent id='defaultLayout'
        cellSpacing={cellSpacing} panels={panels} columns={5} />
      </div>
    </div>
  );
}
export default App;

```

INDEX.JSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

INDEX.TSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

Placing components as content of panels

In a dashboard, components like the chart, grids, maps, gauge, and more can be used to present a complex data. Such components can be placed as the panel content by assigning the corresponding component element as the **content** of the panel.

The following sample demonstrates how to add EJ2 Chart components as the **content** for each panel in the dashboard layout component.

APP.JSX

```
{% raw %}
import { AccumulationChartComponent, AccumulationSeriesCollectionDirective,
AccumulationSeriesDirective, AccumulationTooltip, Category, ChartComponent,
ColumnSeries, DataLabel, Inject, Legend, LineSeries,
SeriesCollectionDirective, SeriesDirective, Tooltip } from "@syncfusion/ej2-
react-charts";
import { DashboardLayoutComponent, PanelDirective, PanelsDirective } from
'@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    const cellSpacing = [10, 10];
    // Template for line Chart
    function lineTemplate() {
        const lineData = [
            { x: 2013, y: 28 }, { x: 2014, y: 25 }, { x: 2015, y: 26 }, { x:
2016, y: 27 },
            { x: 2017, y: 32 }, { x: 2018, y: 35 },
        ];
        return (<div className="template">
            <ChartComponent style={{ "height": "162px" }}><Inject
services={[[LineSeries]]}/>
                <SeriesCollectionDirective>
                    <SeriesDirective dataSource={lineData} xName='x'
yName='y' type='Line' />
                </SeriesCollectionDirective>
            </ChartComponent>
        </div>);
    }
    // Template for Pie Chart
    function pieTemplate() {
        const pieData = [
            { x: 'TypeScript', y: 13, text: 'TS 13%' },
            { x: 'React', y: 12.5, text: 'React 12.5%' },
            { x: 'MVC', y: 12, text: 'MVC 12%' },
            { x: 'Core', y: 12.5, text: 'Core 12.5%' },
            { x: 'Vue', y: 10, text: 'Vue 10%' },
            { x: 'Angular', y: 40, text: 'Angular 40%' }
        ];
        return (<div className="template">
            <AccumulationChartComponent style={{ "height": "162px" }}
tooltip={{ enable: true }}><Inject services={[[AccumulationTooltip]]}/>
                <AccumulationSeriesCollectionDirective>
                    <AccumulationSeriesDirective dataSource={pieData}
xName='x' yName='y' innerRadius="40%" />
                </AccumulationSeriesCollectionDirective>
            </AccumulationChartComponent>
        </div>);
    }
    // Template for Pie Chart 1
    function pieTemplate1() {
        const pieData = [
            { 'x': 'Chrome', y: 37, text: '37%' }, { 'x': 'UC Browser', y:
17, text: '17%' },
            { 'x': 'iPhone', y: 19, text: '19%' },

```

```

        { 'x': 'Others', y: 4, text: '4%' }, { 'x': 'Opera', y: 11, text:
'11%' },
        { 'x': 'Android', y: 12, text: '12%' }
    ];
    const dataLabel = { visible: true, position: 'Inside', name: 'text',
font: { fontWeight: '600' } };
    return (<div className="template">
        <AccumulationChartComponent style={{ "height": "162px" }}
tooltip={{ enable: true }}>
            <Inject services={[AccumulationTooltip]}/>
            <AccumulationSeriesCollectionDirective>
                <AccumulationSeriesDirective dataSource={pieData}
dataLabel={dataLabel} xName='x' yName='y' radius="70%" name='Browser' />
            </AccumulationSeriesCollectionDirective>
        </AccumulationChartComponent>
    </div>);
}
function columnTemplate() {
    const chartData = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    return (<div className="template">
        <ChartComponent style={{ "height": "162px" }} primaryXAxis={{
valueType: 'Category' }}>
            <Inject services={[ColumnSeries, Legend, Tooltip,
Category, DataLabel]}/>
            <SeriesCollectionDirective>
                <SeriesDirective dataSource={chartData} xName='month'
yName='sales' type='Column' />
            </SeriesCollectionDirective>
        </ChartComponent>
    </div>);
}
return (<div>
    <div className="container">
        <div>
            <DashboardLayoutComponent id="dashboard_default" columns={6}
cellSpacing={cellSpacing} allowResizing={true}>
                <PanelsDirective>
                    <PanelDirective sizeX={3} sizeY={2} row={0} col={0}
content={pieTemplate} header="<div>Product usage ratio</div>" />
                    <PanelDirective sizeX={3} sizeY={2} row={0} col={3}
content={columnTemplate} header="<div>Last year Sales Comparison</div>" />
                    <PanelDirective sizeX={3} sizeY={2} row={0} col={3}
content={pieTemplate1} header="<div>Mobile browsers usage</div>" />
                    <PanelDirective sizeX={3} sizeY={2} row={1} col={0}
content={lineTemplate} header="<div>Sales increase percentage</div>" />
                </PanelsDirective>
            </DashboardLayoutComponent>
        </div>
    </div>
</div>);

```

```

}
export default App;
{% enddraw %}

```

APP.TSX

```

{% raw %}
import {
  AccumulationChartComponent, AccumulationSeriesCollectionDirective,
  AccumulationSeriesDirective, AccumulationTooltip, Category,
  ChartComponent, ColumnSeries, DataLabel, Inject, Legend, LineSeries,
  SeriesCollectionDirective, SeriesDirective, Tooltip
} from "@syncfusion/ej2-react-charts";
import { DashboardLayoutComponent, PanelDirective, PanelsDirective } from
"@syncfusion/ej2-react-layouts";
import * as React from 'react';
function App() {
  const cellSpacing: number[] = [10, 10];
  // Template for line Chart
  function lineTemplate(): JSX.Element {
    const lineData: object[] = [
      { x: 2013, y: 28 }, { x: 2014, y: 25 }, { x: 2015, y: 26 }, { x:
2016, y: 27 },
      { x: 2017, y: 32 }, { x: 2018, y: 35 },
    ];
    return (
      <div className="template" >
        <ChartComponent style={{ "height": "162px" }}><Inject
services={[LineSeries]} />
        <SeriesCollectionDirective>
          <SeriesDirective dataSource={lineData} xName='x'
yName='y' type='Line' />
        </SeriesCollectionDirective>
        </ChartComponent>
      </div>
    );
  }
  // Template for Pie Chart
  function pieTemplate(): JSX.Element {
    const pieData: object[] = [
      { x: 'TypeScript', y: 13, text: 'TS 13%' },
      { x: 'React', y: 12.5, text: 'React 12.5%' },
      { x: 'MVC', y: 12, text: 'MVC 12%' },
      { x: 'Core', y: 12.5, text: 'Core 12.5%' },
      { x: 'Vue', y: 10, text: 'Vue 10%' },
      { x: 'Angular', y: 40, text: 'Angular 40%' }
    ];
    return (
      <div className="template" >
        <AccumulationChartComponent style={{ "height": "162px" }}
tooltip={{enable: true}}><Inject services={[ AccumulationTooltip]} />
        <AccumulationSeriesCollectionDirective>
          <AccumulationSeriesDirective dataSource={pieData}
xName='x' yName='y' innerRadius="40%" />
        </AccumulationSeriesCollectionDirective>
        </AccumulationChartComponent>
      </div>
    );
  }
}

```

```

        </div>
    );
}
// Template for Pie Chart 1
function pieTemplate1(): JSX.Element {
    const pieData: object[] = [
        { 'x': 'Chrome', y: 37, text: '37%' }, { 'x': 'UC Browser', y:
17, text: '17%' },
        { 'x': 'iPhone', y: 19, text: '19%' },
        { 'x': 'Others', y: 4, text: '4%' }, { 'x': 'Opera', y: 11, text:
'11%' },
        { 'x': 'Android', y: 12, text: '12%' }
    ];
    const dataLabel: object = { visible: true, position: 'Inside', name:
'text', font: { fontWeight: '600' } };
    return(
        <div className="template" >
            <AccumulationChartComponent style={{ "height": "162px" }}
tooltip={{enable: true}}>
                <Inject services={[ AccumulationTooltip]} />
                <AccumulationSeriesCollectionDirective>
                    <AccumulationSeriesDirective dataSource={pieData}
dataLabel={dataLabel} xName='x' yName='y' radius="70%" name = 'Browser' />
                </AccumulationSeriesCollectionDirective>
            </AccumulationChartComponent>
        </div>
    );
}
function columnTemplate(): JSX.Element {
    const chartData: any[] = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    return(
        <div className="template" >
            <ChartComponent style={{ "height": "162px" }} primaryXAxis={{
valueType: 'Category' }}>
                <Inject services={[ColumnSeries, Legend, Tooltip,
Category, DataLabel]} />
                <SeriesCollectionDirective>
                    <SeriesDirective dataSource={chartData} xName='month'
yName='sales' type='Column' />
                </SeriesCollectionDirective>
            </ChartComponent>
        </div>
    );
}
return (
    <div>
        <div className="container">
            <div>
                <DashboardLayoutComponent id="dashboard_default" columns={6}
cellSpacing={cellSpacing} allowResizing={true}>

```

```

        <PanelsDirective>
          <PanelDirective sizeX={3} sizeY={2} row={0} col={0}
content={pieTemplate as any} header="<div>Product usage ratio</div>" />
          <PanelDirective sizeX={3} sizeY={2} row={0} col={3}
content={columnTemplate as any} header="<div>Last year Sales
Comparison</div>" />
          <PanelDirective sizeX={3} sizeY={2} row={0} col={3}
content={pieTemplate1 as any} header="<div>Mobile browsers usage</div>" />
          <PanelDirective sizeX={3} sizeY={2} row={1} col={0}
content={lineTemplate as any} header="<div>Sales increase percentage</div>" />
        </PanelsDirective>
      </DashboardLayoutComponent>
    </div>
  </div>
</div>
);
}
export default App;
{% enddraw %}

```

INDEX.JSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

INDEX.TSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

You can refer to our [React Dashboard Layout](#) feature tour page for its groundbreaking feature representations. You can also explore our [React Dashboard Layout example](#) to know how to present and manipulate data.

Add remove panels in React Dashboard layout component

In real-time cases, the data being presented within the dashboard should be updated frequently which includes adding or removing the data dynamically within the dashboard. This can be easily achieved by using the `addPanel` and `removePanel` public methods of the component.

Add or remove panels dynamically

Panels can be added dynamically by using the `addPanel` public method by passing the `panel` property as parameter. Also, they can be removed dynamically by using the `removePanel` public method by passing the `panel id` value as a parameter.

It is also possible to remove all the panels in a Dashboard Layout by calling [removeAll](#) method.

```
`js
```

```
dashboard.removeAll();
```

The following sample demonstrates how to add and remove the panels dynamically in the dashboard layout component. Here, panels can be added in any desired position of required size by selecting them in the numeric boxes and clicking add button and remove them by selecting the ID of the panel.

APP.JSX

```
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import { NumericTextBoxComponent } from '@syncfusion/ej2-react-inputs';
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    let count = 7;
    let cellSpacing = [10, 10];
    let resize = ['e-south-east', 'e-east', 'e-west', 'e-north', 'e-south'];
    let panels = [
        { 'id': 'Panel0', 'sizeX': 1, 'sizeY': 1, 'row': 0, 'col': 0,
content: '<div class="content">0</div>' },
        { 'id': 'Panel1', 'sizeX': 3, 'sizeY': 2, 'row': 0, 'col': 1,
content: '<div class="content">1</div>' },
        { 'id': 'Panel2', 'sizeX': 1, 'sizeY': 3, 'row': 0, 'col': 4,
content: '<div class="content">2</div>' },
        { 'id': 'Panel3', 'sizeX': 1, 'sizeY': 1, 'row': 1, 'col': 0,
content: '<div class="content">3</div>' },
        { 'id': 'Panel4', 'sizeX': 2, 'sizeY': 1, 'row': 2, 'col': 0,
content: '<div class="content">4</div>' },
        { 'id': 'Panel5', 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 2,
content: '<div class="content">5</div>' },
        { 'id': 'Panel6', 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 3,
content: '<div class="content">6</div>' }
    ];
    let data = ["Panel0", "Panel1", "Panel2", "Panel3", "Panel4", "Panel5",
"Panel6"];
    let dashboardObj;
    let sizeXObj;
    let sizeYObj;
    let rowObj;
    let colsObj;
    let paneObj;
    // Adding new panels for DashboardLayout
    function onAdd(args) {
        //let proxy = this;
        let panel = [{
            'id': count.toString() + '_layout', 'sizeX': sizeXObj.value,
'sizeY': sizeYObj.value, 'row': rowObj.value, 'col': colsObj.value,
            content: '<div class="content">' + count.toString() +
'</div>'
        }];
        dashboardObj.addPanel(panel[0]);
        count = count + 1;
    }
    // Removeing selected panels for DashboardLayout
    function onRemove(args) {
        dashboardObj.removePanel(paneObj.value.toString());
    }
    return (<div id='container'>
```



```

    <div className="inline" id="control">
      <DashboardLayoutComponent id='dashboard_layout' ref={s =>
        (dashboardObj = s)} cellSpacing={cellSpacing} panels={panels}
        allowResizing={true} columns={5} resizableHandles={resize}/>
    </div>
    <div className="inline" id="properties">
      <table>
        <tbody>
          <tr>
            <td>SizeX</td>
            <td> <NumericTextBoxComponent ref={s => (sizeXObj = s)}
              className="col-sm-4" placeholder={"Ex: 10"} value={1} min={1} max={5}
              floatLabelType="Never" id="sizeX"/></td>
          </tr>
          <tr>
            <td>SizeY</td>
            <td> <NumericTextBoxComponent ref={s => (sizeYObj = s)}
              className="col-sm-4" placeholder={"Ex: 10"} value={1} min={1} max={5}
              floatLabelType="Never" id="sizeY"/></td>
          </tr>
          <tr>
            <td>Row</td>
            <td> <NumericTextBoxComponent ref={s => (rowObj = s)}
              className="col-sm-4" placeholder={"Ex: 10"} value={0} min={0} max={5}
              floatLabelType="Never" id="row"/></td>
          </tr>
          <tr>
            <td>Column</td>
            <td> <NumericTextBoxComponent ref={s => (colsObj = s)}
              className="col-sm-4" placeholder={"Ex: 10"} value={0} min={0} max={4}
              floatLabelType="Never" id="column"/></td>
          </tr>
          <tr>
            <td />
            <td>
              <button onClick={onAdd}>Add Panel</button>
            </td>
          </tr>
        </tbody>
      </table>
      <table>
        <tbody>
          <tr>
            <td>Id</td>
            <td> <DropDownListComponent ref={s => (paneObj = s)}
              className="col-sm-4" placeholder={"panel id"} dataSource={data}
              floatLabelType="Never" id="panel_id"/></td>
          </tr>
          <tr>
            <td />
            <td>
              <button onClick={onRemove}>Remove Panel</button>
            </td>
          </tr>
        </tbody>
      </table>
    </div>

```

```

    </div>);
}
export default App;

```

APP.TSX

```

import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import { NumericTextBoxComponent } from '@syncfusion/ej2-react-inputs';
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    let count = 7;
    let cellSpacing = [10, 10];
    let resize = ['e-south-east', 'e-east', 'e-west', 'e-north', 'e-south'];
    let panels = [
        { 'id': 'Panel0', 'sizeX': 1, 'sizeY': 1, 'row': 0, 'col': 0,
content: '<div class="content">0</div>' },
        { 'id': 'Panel1', 'sizeX': 3, 'sizeY': 2, 'row': 0, 'col': 1,
content: '<div class="content">1</div>' },
        { 'id': 'Panel2', 'sizeX': 1, 'sizeY': 3, 'row': 0, 'col': 4,
content: '<div class="content">2</div>' },
        { 'id': 'Panel3', 'sizeX': 1, 'sizeY': 1, 'row': 1, 'col': 0,
content: '<div class="content">3</div>' },
        { 'id': 'Panel4', 'sizeX': 2, 'sizeY': 1, 'row': 2, 'col': 0,
content: '<div class="content">4</div>' },
        { 'id': 'Panel5', 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 2,
content: '<div class="content">5</div>' },
        { 'id': 'Panel6', 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 3,
content: '<div class="content">6</div>' }
    ];
    let data = ["Panel0", "Panel1", "Panel2", "Panel3", "Panel4", "Panel5",
"Panel6"];
    let dashboardObj:DashboardLayoutComponent;
    let sizeXObj:NumericTextBoxComponent;
    let sizeYObj:NumericTextBoxComponent;
    let rowObj:NumericTextBoxComponent;
    let colsObj:NumericTextBoxComponent;
    let paneObj:DropDownListComponent;
    // Adding new panels for DashboardLayout
    function onAdd(args) {
        //let proxy = this;
        let panel = [{
            'id': count.toString() + '_layout', 'sizeX': sizeXObj.value,
'sizeY': sizeYObj.value, 'row': rowObj.value, 'col': colsObj.value,
            content: '<div class="content">' + count.toString() +
'</div>'
        }];
        (dashboardObj as any).addPanel(panel[0]);
        count = count + 1;
    }
    // Removeing selected panels for DashboardLayout
    function onRemove(args) {
        (dashboardObj as any).removePanel(paneObj.value.toString());
    }
    return (<div id='container'>
        <div className="inline" id="control">

```

```

    <DashboardLayoutComponent id='dashboard_layout' ref={s =>
    (dashboardObj = s)} cellSpacing={cellSpacing} panels={panels}
    allowResizing={true} columns={5} resizableHandles={resize}/>
  </div>
  <div className="inline" id="properties">
    <table>
      <tbody>
        <tr>
          <td>SizeX</td>
          <td> <NumericTextBoxComponent ref={s => (sizeXObj = s)}
className="col-sm-4" placeholder={"Ex: 10"} value={1} min={1} max={5}
floatLabelType="Never" id="sizeX"/></td>
        </tr>
        <tr>
          <td>SizeY</td>
          <td> <NumericTextBoxComponent ref={s => (sizeYObj = s)}
className="col-sm-4" placeholder={"Ex: 10"} value={1} min={1} max={5}
floatLabelType="Never" id="sizeY"/></td>
        </tr>
        <tr>
          <td>Row</td>
          <td> <NumericTextBoxComponent ref={s => (rowObj = s)}
className="col-sm-4" placeholder={"Ex: 10"} value={0} min={0} max={5}
floatLabelType="Never" id="row"/></td>
        </tr>
        <tr>
          <td>Column</td>
          <td> <NumericTextBoxComponent ref={s => (colsObj = s)}
className="col-sm-4" placeholder={"Ex: 10"} value={0} min={0} max={4}
floatLabelType="Never" id="column"/></td>
        </tr>
        <tr>
          <td />
          <td>
            <button onClick={onAdd}>Add Panel</button>
          </td>
        </tr>
      </tbody>
    </table>
    <table>
      <tbody>
        <tr>
          <td>Id</td>
          <td> <DropDownListComponent ref={s => (paneObj = s)}
className="col-sm-4" placeholder={"panel id"} dataSource={data}
floatLabelType="Never" id="panel_id"/></td>
        </tr>
        <tr>
          <td />
          <td>
            <button onClick={onRemove}>Remove Panel</button>
          </td>
        </tr>
      </tbody>
    </table>
  </div>
</div>);

```

```
}  
export default App;
```

INDEX.JSX

```
import * as React from 'react';  
import * as ReactDOM from 'react-dom';  
import App from './App';  
ReactDOM.render(<App />, document.getElementById('root'));
```

INDEX.TSX

```
import * as React from 'react';  
import * as ReactDOM from 'react-dom';  
import App from './App';  
ReactDOM.render(<App />, document.getElementById('root'));
```

You can refer to our [React Dashboard Layout](#) feature tour page for its groundbreaking feature representations. You can also explore our [React Dashboard Layout example](#) to know how to present and manipulate data.

Interaction With Panels

Dragging moving of panels in React Dashboard layout component

The Dashboard Layout component is provided with dragging functionality to drag and reorder the panels within the layout. While dragging a panel, a holder will be highlighted below the panel indicating the panel placement on panel drop. This helps the user to decide whether to place the panel in the current position or revert to previous position without disturbing the layout.

If one or more panels collide while dragging, then the colliding panels will be pushed towards the left or right or top or bottom direction where an adaptive space for the collided panel is available. The position changes of these collided panels will be updated dynamically during dragging of a panel, so the user can conclude whether to place the panel in the current position or not.

While dragging a panel in Dashboard layout the following dragging events will be triggered,

- [dragStart](#) - Triggers when panel drag starts
- [drag](#) - Triggers when panel is being dragged
- [dragStop](#) - Triggers when panel drag stops

The following sample demonstrates dragging and pushing of panels. For example, while dragging the panel 0 over panel 1, these panels get collided and push the panel 1 towards the feasible direction, so that, the panel 0 gets placed in the panel 1 position.

APP.JSX

```
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';  
import * as React from 'react';  
function App() {  
  const cellSpacing = [10, 10];  
  let panels = [  
    { 'sizeX': 1, 'sizeY': 1, 'row': 0, 'col': 0, content: '<div  
class="content">0</div>' },
```

```

    { 'sizeX': 3, 'sizeY': 2, 'row': 0, 'col': 1, content: '<div
class="content">1</div>' },
    { 'sizeX': 1, 'sizeY': 3, 'row': 0, 'col': 4, content: '<div
class="content">2</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 1, 'col': 0, content: '<div
class="content">3</div>' },
    { 'sizeX': 2, 'sizeY': 1, 'row': 2, 'col': 0, content: '<div
class="content">4</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 2, content: '<div
class="content">5</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 3, content: '<div
class="content">6</div>' }
  ];
  function onDragStart() {
    console.log("Drag start");
  }
  //Dashboard Layout's drag event function
  function onDrag(args) {
    console.log("Dragging");
  }
  //Dashboard Layout's dragstop event function
  function onDragStop(args) {
    console.log("Drag stop");
  }
  return (<div>
    <div className="container">
      <DashboardLayoutComponent id='defaultLayout' cellSpacing={cellSpacing}
panels={panels} columns={5} dragStart={onDragStart.bind(this)}
drag={onDrag.bind(this)} dragStop={onDragStop.bind(this)} />
    </div>
  </div>);
}
export default App;

```

APP.TSX

```

import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
  const cellSpacing: number[] = [10, 10];
  let panels: any = [
    { 'sizeX': 1, 'sizeY': 1, 'row': 0, 'col': 0, content: '<div
class="content">0</div>' },
    { 'sizeX': 3, 'sizeY': 2, 'row': 0, 'col': 1, content: '<div
class="content">1</div>' },
    { 'sizeX': 1, 'sizeY': 3, 'row': 0, 'col': 4, content: '<div
class="content">2</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 1, 'col': 0, content: '<div
class="content">3</div>' },
    { 'sizeX': 2, 'sizeY': 1, 'row': 2, 'col': 0, content: '<div
class="content">4</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 2, content: '<div
class="content">5</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 3, content: '<div
class="content">6</div>' }
  ];
}

```

```
function onDragStart() {
  console.log("Drag start");
}
//Dashboard Layout's drag event function
function onDrag(args: any) {
  console.log("Dragging");
}
//Dashboard Layout's dragstop event function
function onDragStop(args: any) {
  console.log("Drag stop");
}
return (
  <div>
    <div className="container">
      <DashboardLayoutComponent id='defaultLayout' cellSpacing={cellSpacing}
panels={panels} columns={5}
      dragStart={onDragStart.bind(this)} drag={onDrag.bind(this)}
      dragStop={onDragStop.bind(this)} />
    </div>
  </div>
);
}
export default App;
```

INDEX.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));
```

INDEX.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));
```

Customizing the dragging handler

Initially, the complete panel will act as the handler for dragging the panel such that the dragging action occurs on clicking anywhere over a panel. However, this dragging handler for the panels can be customized using the `draggableHandle` property to restrict the dragging action within a particular element in the panel.

The following sample demonstrates customizing the dragging handler of the panels where the dragging action of panel occurs only with the header of the panel.

APP.JSX

```
{% raw %}
import { AccumulationChartComponent, AccumulationSeriesCollectionDirective,
AccumulationSeriesDirective, AccumulationTooltip, Category, ChartComponent,
ColumnSeries, DataLabel, Inject, Legend, LineSeries,
SeriesCollectionDirective, SeriesDirective, Tooltip } from "@syncfusion/ej2-
react-charts";
```

```

import { DashboardLayoutComponent, PanelDirective, PanelsDirective } from
'@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    const cellSpacing = [10, 10];
    // Template for line Chart
    function lineTemplate() {
        const lineData = [
            { x: 2013, y: 28 }, { x: 2014, y: 25 }, { x: 2015, y: 26 }, { x:
2016, y: 27 },
            { x: 2017, y: 32 }, { x: 2018, y: 35 },
        ];
        return (<div className="template">
            <ChartComponent style={{ "height": "162px" }}>
                <Inject services={[LineSeries]}>
                    <SeriesCollectionDirective>
                        <SeriesDirective dataSource={lineData} xName='x'
yName='y' type='Line'>
                    </SeriesCollectionDirective>
                </ChartComponent>
            </div>);
    }
    // Template for Pie Chart
    function pieTemplate() {
        const pieData = [
            { x: 'TypeScript', y: 13, text: 'TS 13%' },
            { x: 'React', y: 12.5, text: 'React 12.5%' },
            { x: 'MVC', y: 12, text: 'MVC 12%' },
            { x: 'Core', y: 12.5, text: 'Core 12.5%' },
            { x: 'Vue', y: 10, text: 'Vue 10%' },
            { x: 'Angular', y: 40, text: 'Angular 40%' }
        ];
        return (<div className="template">
            <AccumulationChartComponent style={{ "height": "162px" }}
tooltip={{ enable: true }}>
                <Inject services={[AccumulationTooltip]}>
                    <AccumulationSeriesCollectionDirective>
                        <AccumulationSeriesDirective dataSource={pieData}
xName='x' yName='y' innerRadius="40%">
                    </AccumulationSeriesCollectionDirective>
                </AccumulationChartComponent>
            </div>);
    }
    // Template for Pie Chart 1
    function pieTemplate1() {
        const pieData = [
            { 'x': 'Chrome', y: 37, text: '37%' },
            { 'x': 'UC Browser', y: 17, text: '17%' },
            { 'x': 'iPhone', y: 19, text: '19%' },
            { 'x': 'Others', y: 4, text: '4%' },
            { 'x': 'Opera', y: 11, text: '11%' },
            { 'x': 'Android', y: 12, text: '12%' }
        ];
        const dataLabel = { visible: true, position: 'Inside', name: 'text',
font: { fontWeight: '600' } };
        const enableAnimation = false;
        return (<div className="template">

```

```

        <AccumulationChartComponent style={{ "height": "162px" }}
enableAnimation={enableAnimation} legendSettings={{ visible: false }}
tooltip={{ enable: true, format: '${point.x} : <b>${point.y}%</b>' }}>
        <Inject services={[AccumulationTooltip]}/>
        <AccumulationSeriesCollectionDirective>
            <AccumulationSeriesDirective dataSource={pieData}
dataLabel={dataLabel} xName='x' yName='y' radius="70%" name='Browser' />
        </AccumulationSeriesCollectionDirective>
    </AccumulationChartComponent>
</div>;
}
function columnTemplate() {
    const chartData = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    return (<div className="template">
        <ChartComponent style={{ "height": "162px" }} primaryXAxis={{
valueType: 'Category' }}>
            <Inject services={[ColumnSeries, Legend, Tooltip,
Category, DataLabel]}/>
            <SeriesCollectionDirective>
                <SeriesDirective dataSource={chartData}
xName='month' yName='sales' type='Column' />
            </SeriesCollectionDirective>
        </ChartComponent>
    </div>;
}
return (<div id='container'>
    <DashboardLayoutComponent id="dashboard_default"
draggableHandle='.e-panel-header' columns={6} cellSpacing={cellSpacing}
allowResizing={true}>
        <PanelsDirective>
            <PanelDirective sizeX={3} sizeY={2} row={0} col={0}
content={pieTemplate} header="<div class='header'>Product usage
ratio</div><span class='handler e-icons burg-icon'></span>" />
            <PanelDirective sizeX={3} sizeY={2} row={0} col={3}
content={columnTemplate} header="<div class='header'>Last year Sales
Comparison</div><span class='handler e-icons burg-icon'></span>" />
            <PanelDirective sizeX={3} sizeY={2} row={0} col={3}
content={pieTemplate1} header="<div class='header'>Mobile browsers
usage</div><span class='handler e-icons burg-icon'></span>" />
            <PanelDirective sizeX={3} sizeY={2} row={1} col={0}
content={lineTemplate} header="<div class='header'>Sales increase
percentage</div><span class='handler e-icons burg-icon'></span>" />
        </PanelsDirective>
    </DashboardLayoutComponent>
</div>;
}
export default App;
{% endraw %}

```


APP.TSX

```
{% raw %}
import {
    AccumulationChartComponent, AccumulationSeriesCollectionDirective,
    AccumulationSeriesDirective, AccumulationTooltip, Category,
    ChartComponent, ColumnSeries, DataLabel, Inject, Legend, LineSeries,
    SeriesCollectionDirective, SeriesDirective, Tooltip
} from "@syncfusion/ej2-react-charts";
import { DashboardLayoutComponent, PanelDirective, PanelsDirective } from
"@syncfusion/ej2-react-layouts";
import * as React from 'react';
function App() {
    const cellSpacing: number[] = [10, 10];
    // Template for line Chart
    function lineTemplate(): JSX.Element {
        const lineData: object[] = [
            { x: 2013, y: 28 }, { x: 2014, y: 25 }, { x: 2015, y: 26 }, { x:
2016, y: 27 },
            { x: 2017, y: 32 }, { x: 2018, y: 35 },
        ];
        return (
            <div className="template" >
                <ChartComponent style={{ "height": "162px" }} >
                    <Inject services={[LineSeries]} />
                    <SeriesCollectionDirective>
                        <SeriesDirective dataSource={lineData} xName='x'
yName='y' type='Line' />
                    </SeriesCollectionDirective>
                </ChartComponent>
            </div>
        );
    }
    // Template for Pie Chart
    function pieTemplate(): JSX.Element {
        const pieData: object[] = [
            { x: 'TypeScript', y: 13, text: 'TS 13%' },
            { x: 'React', y: 12.5, text: 'React 12.5%' },
            { x: 'MVC', y: 12, text: 'MVC 12%' },
            { x: 'Core', y: 12.5, text: 'Core 12.5%' },
            { x: 'Vue', y: 10, text: 'Vue 10%' },
            { x: 'Angular', y: 40, text: 'Angular 40%' }
        ];
        return (
            <div className="template" >
                <AccumulationChartComponent style={{ "height": "162px" }}
tooltip={{enable: true}} >
                    <Inject services={[ AccumulationTooltip]} />
                    <AccumulationSeriesCollectionDirective>
                        <AccumulationSeriesDirective dataSource={pieData}
xName='x' yName='y' innerRadius="40%" />
                    </AccumulationSeriesCollectionDirective>
                </AccumulationChartComponent>
            </div>
        );
    }
}
// Template for Pie Chart 1
```

```

function pieTemplate1(): JSX.Element {
    const pieData: object[] = [
        { 'x': 'Chrome', y: 37, text: '37%' },
        { 'x': 'UC Browser', y: 17, text: '17%' },
        { 'x': 'iPhone', y: 19, text: '19%' },
        { 'x': 'Others', y: 4, text: '4%' },
        { 'x': 'Opera', y: 11, text: '11%' },
        { 'x': 'Android', y: 12, text: '12%' }
    ];
    const dataLabel: object = { visible: true, position: 'Inside', name:
'text', font: { fontWeight: '600' } };
    const enableAnimation: boolean = false;
    return(
        <div className="template" >
            <AccumulationChartComponent style={{ "height": "162px" }}
enableAnimation={ enableAnimation } legendSettings= {{ visible: false }}
                tooltip={{enable: true, format: '${point.x} :
<b>${point.y}%</b>' }}>
                    <Inject services={[ AccumulationTooltip] } />
                    <AccumulationSeriesCollectionDirective>
                        <AccumulationSeriesDirective dataSource={pieData}
dataLabel={dataLabel} xName='x' yName='y' radius="70%" name = 'Browser' />
                    </AccumulationSeriesCollectionDirective>
                </AccumulationChartComponent>
            </div>
        );
}
function columnTemplate(): JSX.Element {
    const chartData: any[] = [
        { month: 'Jan', sales: 35 }, { month: 'Feb', sales: 28 },
        { month: 'Mar', sales: 34 }, { month: 'Apr', sales: 32 },
        { month: 'May', sales: 40 }, { month: 'Jun', sales: 32 },
        { month: 'Jul', sales: 35 }, { month: 'Aug', sales: 55 },
        { month: 'Sep', sales: 38 }, { month: 'Oct', sales: 30 },
        { month: 'Nov', sales: 25 }, { month: 'Dec', sales: 32 }
    ];
    return(
        <div className="template" >
            <ChartComponent style={{ "height": "162px" }} primaryXAxis={{
valueType: 'Category' }}>
                <Inject services={[ColumnSeries, Legend, Tooltip,
Category, DataLabel] } />
                <SeriesCollectionDirective>
                    <SeriesDirective dataSource={chartData}
xName='month' yName='sales' type='Column' />
                </SeriesCollectionDirective>
            </ChartComponent>
        </div>
    );
}
return (
    <div id='container'>
        <DashboardLayoutComponent id="dashboard_default"
draggableHandle='.e-panel-header' columns={6} cellSpacing={cellSpacing}
allowResizing={true}>
            <PanelsDirective>

```

```

        <PanelDirective sizeX={3} sizeY={2} row={0} col={0}
content={pieTemplate as any} header="<div class='header'>Product usage
ratio</div><span class='handler e-icons burg-icon'></span>" />
        <PanelDirective sizeX={3} sizeY={2} row={0} col={3}
content={columnTemplate as any} header="<div class='header'>Last year Sales
Comparison</div><span class='handler e-icons burg-icon'></span>" />
        <PanelDirective sizeX={3} sizeY={2} row={0} col={3}
content={pieTemplate1 as any} header="<div class='header'>Mobile browsers
usage</div><span class='handler e-icons burg-icon'></span>" />
        <PanelDirective sizeX={3} sizeY={2} row={1} col={0}
content={lineTemplate as any} header="<div class='header'>Sales increase
percentage</div><span class='handler e-icons burg-icon'></span>" />
    </PanelsDirective>
  </DashboardLayoutComponent>
</div>
);
}
export default App;
{% enddraw %}

```

INDEX.JSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

INDEX.TSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

Disable dragging of panels

By default, the dragging of panels is enabled in Dashboard Layout. It can also be disabled with the help of [allowDragging](#) API. Setting [allowDragging](#) to false disables the dragging functionality in Dashboard Layout.

The following sample demonstrates Dashboard Layout with dragging support disabled.

APP.JSX

```

import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    const cellSpacing = [10, 10];
    let allowDragging = false;
    let panels = [
        { 'sizeX': 1, 'sizeY': 1, 'row': 0, 'col': 0, content: '<div
class="content">0</div>' },
        { 'sizeX': 3, 'sizeY': 2, 'row': 0, 'col': 1, content: '<div
class="content">1</div>' },
        { 'sizeX': 1, 'sizeY': 3, 'row': 0, 'col': 4, content: '<div
class="content">2</div>' },

```

```

        { 'sizeX': 1, 'sizeY': 1, 'row': 1, 'col': 0, content: '<div
class="content">3</div>' },
        { 'sizeX': 2, 'sizeY': 1, 'row': 2, 'col': 0, content: '<div
class="content">4</div>' },
        { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 2, content: '<div
class="content">5</div>' },
        { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 3, content: '<div
class="content">6</div>' }
    ];
    return (<div>
        <div className="container">
            <DashboardLayoutComponent id='defaultLayout'
cellSpacing={cellSpacing} panels={panels} columns={5}
allowDragging={allowDragging}/>
        </div>
    </div>);
}
export default App;

```

APP.TSX

```

import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    const cellSpacing: number[] = [10, 10];
    let allowDragging: boolean = false;
    let panels: any = [
        { 'sizeX': 1, 'sizeY': 1, 'row': 0, 'col': 0, content: '<div
class="content">0</div>' },
        { 'sizeX': 3, 'sizeY': 2, 'row': 0, 'col': 1, content: '<div
class="content">1</div>' },
        { 'sizeX': 1, 'sizeY': 3, 'row': 0, 'col': 4, content: '<div
class="content">2</div>' },
        { 'sizeX': 1, 'sizeY': 1, 'row': 1, 'col': 0, content: '<div
class="content">3</div>' },
        { 'sizeX': 2, 'sizeY': 1, 'row': 2, 'col': 0, content: '<div
class="content">4</div>' },
        { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 2, content: '<div
class="content">5</div>' },
        { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 3, content: '<div
class="content">6</div>' }
    ];
    return (
        <div>
            <div className="container">
                <DashboardLayoutComponent id='defaultLayout'
cellSpacing={cellSpacing} panels={panels} columns={5}
allowDragging={allowDragging} />
            </div>
        </div>
    );
}
export default App;

```

INDEX.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));
```

INDEX.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));
```

You can refer to our [React Dashboard Layout](#) feature tour page for its groundbreaking feature representations. You can also explore our [React Dashboard Layout example](#) to know how to present and manipulate data.

Moving panels in React Dashboard layout component

Other than drag and drop, it is possible to move the panels in Dashboard Layout programmatically. This can be achieved using [movePanel](#) method. The method is invoked as follows,

```
`js
```

```
movePanel(id, row, col)
```

```
,
```

Where,

- id - ID of the panel which needs to be moved.
- row - New row position for moving the panel.
- col - New column position for moving the panel.

Each time a panel's position is changed (Programmatically or through UI interaction), the Dashboard Layout's [change](#) event will be triggered.

The following sample demonstrates moving a panel programmatically to a new position in the Dashboard Layout's [created](#) event.

APP.JSX

```
{% raw %}
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
  const cellSpacing = [10, 10];
  let panels = [
    { 'sizeX': 1, 'sizeY': 1, 'row': 0, 'col': 0, content: '<div class="content">0</div>' },
    { 'sizeX': 3, 'sizeY': 2, 'row': 0, 'col': 1, content: '<div class="content">1</div>' },
    { 'sizeX': 1, 'sizeY': 3, 'row': 0, 'col': 4, content: '<div class="content">2</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 1, 'col': 0, content: '<div class="content">3</div>' },
  ],
```

```

    { 'sizeX': 2, 'sizeY': 1, 'row': 2, 'col': 0, content: '<div
class="content">4</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 2, content: '<div
class="content">5</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 3, content: '<div
class="content">6</div>' }
  ];
  //Dashboard Layout's created event function
  function onCreate(args) {
    // movePanel("id", row, col)
    dashboardObj.movePanel("layout_0", 1, 0);
  }
  //Dashboard Layout's change event function
  function onChange(args) {
    console.log("Change event triggered");
  }
  let dashboardObj;
  return (<div>
    <div className="container">
      <DashboardLayoutComponent id='defaultLayout' ref={(scope) => {
dashboardObj = scope; }} cellSpacing={cellSpacing} panels={panels}
columns={5} created={onCreate.bind(this)} change={onChange.bind(this)} />
    </div>
  </div>);
}
export default App;
{% endraw %}

```

APP.TSX

```

{% raw %}
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
  const cellSpacing: number[] = [10, 10];
  let panels: any = [
    { 'sizeX': 1, 'sizeY': 1, 'row': 0, 'col': 0, content: '<div
class="content">0</div>' },
    { 'sizeX': 3, 'sizeY': 2, 'row': 0, 'col': 1, content: '<div
class="content">1</div>' },
    { 'sizeX': 1, 'sizeY': 3, 'row': 0, 'col': 4, content: '<div
class="content">2</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 1, 'col': 0, content: '<div
class="content">3</div>' },
    { 'sizeX': 2, 'sizeY': 1, 'row': 2, 'col': 0, content: '<div
class="content">4</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 2, content: '<div
class="content">5</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 3, content: '<div
class="content">6</div>' }
  ];
  //Dashboard Layout's created event function
  function onCreate(args: any) {
    // movePanel("id", row, col)
    dashboardObj.movePanel("layout_0", 1, 0);
  }
}

```

```
//Dashboard Layout's change event function
function onChange(args: any) {
    console.log("Change event triggered");
}
let dashboardObj: DashboardLayoutComponent;
return (
    <div>
        <div className="container">
            <DashboardLayoutComponent id='defaultLayout' ref={(scope) => {
                dashboardObj = scope; }} cellSpacing={cellSpacing}
                panels={panels} columns={5} created={onCreated.bind(this)}
                change={onChange.bind(this)} />
            </div>
        </div>
    );
}
export default App;
{% enddraw %}
```

INDEX.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));
```

INDEX.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));
```

You can refer to our [React Dashboard Layout](#) feature tour page for its groundbreaking feature representations. You can also explore our [React Dashboard Layout example](#) to know how to present and manipulate data.

Resizing of panels in React Dashboard layout component

The Dashboard Layout component is also provided with the panel resizing functionality which can be enabled or disabled using the `allowResizing` property. This functionality allows you to resize the panels dynamically through UI interactions using the resizing handlers which controls the panel resizing in various directions.

Initially, the panels can be resized only in south-east direction. However, panels can also be resized in east, west, north, south and south-west directions by defining the required directions with the `resizableHandles` property.

On resizing a panel in Dashboard layout the following events will be triggered,

- [resizeStart](#) - Triggers when panel resize starts
- [resize](#) - Triggers when panel is being resized
- [resizeStop](#) - Triggers when panel resize stops

The following sample demonstrates how to enable and disable the resizing of panels in the Dashboard Layout component in different directions.

APP.JSX

```
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    const cellSpacing = [10, 10];
    let resize = ['e-south-east', 'e-east', 'e-west', 'e-north', 'e-south'];
    let panels = [
        { 'sizeX': 1, 'sizeY': 1, 'row': 0, 'col': 0, content: '<div class="content">0</div>' },
        { 'sizeX': 3, 'sizeY': 2, 'row': 0, 'col': 1, content: '<div class="content">1</div>' },
        { 'sizeX': 1, 'sizeY': 3, 'row': 0, 'col': 4, content: '<div class="content">2</div>' },
        { 'sizeX': 1, 'sizeY': 1, 'row': 1, 'col': 0, content: '<div class="content">3</div>' },
        { 'sizeX': 2, 'sizeY': 1, 'row': 2, 'col': 0, content: '<div class="content">4</div>' },
        { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 2, content: '<div class="content">5</div>' },
        { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 3, content: '<div class="content">6</div>' }
    ];
    function onResizeStart() {
        console.log("Resize start");
    }
    //Dashboard Layout's drag event function
    function onResize(args) {
        console.log("Resizing");
    }
    //Dashboard Layout's dragstop event function
    function onResizeStop(args) {
        console.log("Resize stop");
    }
    return (
        <div>
            <div id="container">
                <DashboardLayoutComponent id='defaultLayout'
                    cellSpacing={cellSpacing} panels={panels} allowResizing={true} columns={5}
                    resizableHandles={resize} resizeStart={onResizeStart.bind(this)}
                    resize={onResize.bind(this)} resizeStop={onResizeStop.bind(this)} />
            </div>
        </div>);
}
export default App;
```

APP.TSX

```
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    const cellSpacing: number[] = [10, 10];
    let resize: any[] = ['e-south-east', 'e-east', 'e-west', 'e-north', 'e-south'];
    let panels: any = [
```



```

    {'sizeX': 1, 'sizeY': 1, 'row': 0, 'col': 0, content: '<div
class="content">0</div>'},
    {'sizeX': 3, 'sizeY': 2, 'row': 0, 'col': 1, content: '<div
class="content">1</div>'},
    {'sizeX': 1, 'sizeY': 3, 'row': 0, 'col': 4, content: '<div
class="content">2</div>'},
    {'sizeX': 1, 'sizeY': 1, 'row': 1, 'col': 0, content: '<div
class="content">3</div>'},
    {'sizeX': 2, 'sizeY': 1, 'row': 2, 'col': 0, content: '<div
class="content">4</div>'},
    {'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 2, content: '<div
class="content">5</div>'},
    {'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 3, content: '<div
class="content">6</div>'}
  ];
  function onResizeStart() {
    console.log("Resize start");
  }
  //Dashboard Layout's drag event function
  function onResize(args: any) {
    console.log("Resizing");
  }
  //Dashboard Layout's dragstop event function
  function onResizeStop(args: any) {
    console.log("Resize stop");
  }
  return (
    <div>
      <div id="container">
        <DashboardLayoutComponent id='defaultLayout'
cellSpacing={cellSpacing} panels={panels} allowResizing={true} columns={5}
resizableHandles={resize}
          resizeStart={onResizeStart.bind(this)}
resize={onResize.bind(this)} resizeStop={onResizeStop.bind(this)} />
      </div>
    </div>
  );
}
export default App;

```

INDEX.JSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

INDEX.TSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

Resizing panels programmatically

The Dashboard Layout panels can also be resized programmatically by using [resizePanel](#) method. The method is invoked as follows,

```
`js
```

```
resizePanel(id, sizeX, sizeY)
```

Where,

- id - ID of the panel which needs to be resized.
- sizeX - New panel width in cells count for resizing the panel.
- sizeY - New panel height in cells count for resizing the panel.

The following sample demonstrates resizing panels programmatically in the Dashboard Layout's [created](#) event.

APP.JSX

```
{% raw %}
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
  const cellSpacing = [10, 10];
  let panels = [
    { 'sizeX': 1, 'sizeY': 1, 'row': 0, 'col': 0, content: '<div
class="content">0</div>' },
    { 'sizeX': 3, 'sizeY': 2, 'row': 0, 'col': 1, content: '<div
class="content">1</div>' },
    { 'sizeX': 1, 'sizeY': 3, 'row': 0, 'col': 4, content: '<div
class="content">2</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 1, 'col': 0, content: '<div
class="content">3</div>' },
    { 'sizeX': 2, 'sizeY': 1, 'row': 2, 'col': 0, content: '<div
class="content">4</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 2, content: '<div
class="content">5</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 3, content: '<div
class="content">6</div>' }
  ];
  //Dashboard Layout's created event function
  function onCreated(args) {
    // resizePanel("id", sizeX, sizeY)
    dashboardObj.resizePanel("layout_4", 1, 1);
    dashboardObj.resizePanel("layout_5", 2, 1);
  }
  let dashboardObj;
  return (<div>
    <div className="container">
      <DashboardLayoutComponent id='defaultLayout' ref={(scope) => {
dashboardObj = scope; }} cellSpacing={cellSpacing} panels={panels}
columns={5} created={onCreated.bind(this)} />
    </div>
  </div>);
}
```

```
export default App;
{% enddraw %}
```

APP.TSX

```
{% raw %}
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
  const cellSpacing: number[] = [10, 10];
  let panels: any = [
    { 'sizeX': 1, 'sizeY': 1, 'row': 0, 'col': 0, content: '<div
class="content">0</div>' },
    { 'sizeX': 3, 'sizeY': 2, 'row': 0, 'col': 1, content: '<div
class="content">1</div>' },
    { 'sizeX': 1, 'sizeY': 3, 'row': 0, 'col': 4, content: '<div
class="content">2</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 1, 'col': 0, content: '<div
class="content">3</div>' },
    { 'sizeX': 2, 'sizeY': 1, 'row': 2, 'col': 0, content: '<div
class="content">4</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 2, content: '<div
class="content">5</div>' },
    { 'sizeX': 1, 'sizeY': 1, 'row': 2, 'col': 3, content: '<div
class="content">6</div>' }
  ];
  //Dashboard Layout's created event function
  function onCreated(args: any) {
    // resizePanel("id", sizeX, sizeY)
    dashboardObj.resizePanel("layout_4", 1, 1);
    dashboardObj.resizePanel("layout_5", 2, 1);
  }
  let dashboardObj: DashboardLayoutComponent;
  return (
    <div>
      <div className="container">
        <DashboardLayoutComponent id='defaultLayout' ref={(scope) => {
dashboardObj = scope; }} cellSpacing={cellSpacing}
          panels={panels} columns={5} created={onCreated.bind(this)} />
        </div>
      </div>
    </div>
  );
}
export default App;
{% enddraw %}
```

INDEX.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));
```

INDEX.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));
```

You can refer to our [React Dashboard Layout](#) feature tour page for its groundbreaking feature representations. You can also explore our [React Dashboard Layout example](#) to know how to present and manipulate data.

Floating of panels in React Dashboard layout component

The floating functionality of the component allows you to effectively use the entire layout for the panel's placement. If the floating functionality is enabled, the panels within the layout get floated upwards automatically to occupy the empty cells available in previous rows. This functionality can be enabled or disabled using the `allowFloating` property of the component.

The following sample demonstrates how to enable or disable the floating of panels in the Dashboard Layout component.

APP.JSX

```
import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    let dashboardObj;
    let buttonObj;
    let resetPanels = [];
    const cellSpacing = [10, 10];
    let panels = [
        { 'sizeX': 2, 'sizeY': 2, 'row': 1, 'col': 0, content: '<div class="content">0</div>' },
        { 'sizeX': 2, 'sizeY': 2, 'row': 2, 'col': 2, content: '<div class="content">1</div>' },
        { 'sizeX': 2, 'sizeY': 2, 'row': 3, 'col': 4, content: '<div class="content">2</div>' }
    ];
    // Button click action to change the floating options for DashboardLayout
    function onClick(args) {
        if (buttonObj.value === "Disable Floating and Reset") {
            buttonObj.value = 'Enable Floating';
            dashboardObj.allowFloating = false;
            dashboardObj.panels = resetPanels;
        }
        else {
            buttonObj.value = 'Disable Floating and Reset';
            dashboardObj.allowFloating = true;
        }
    }
    // Using created event to initialize the panels
    function onCreate(args) {
        resetPanels = dashboardObj.serialize();
        resetPanels[0].content = '<div class="content">0</div>';
        resetPanels[1].content = '<div class="content">1</div>';
        resetPanels[2].content = '<div class="content">2</div>';
    }
    return (<div>
```

```

        <div id='container'>
          <div className="inline" id="control">
            <DashboardLayoutComponent id='defaultLayout' ref={s =>
(dashboardObj = s)} cellSpacing={cellSpacing} created={onCreate}
panels={panels} allowFloating={false} cellAspectRatio={100 / 75}
columns={6}/>
          </div>
          <div className="inline" id="properties">
            <input type="button" ref={s => buttonObj = s} className="e-btn
e-flat e-primary e-outline" name="floating" id="floating" value="Enable
Floating" onClick={onClick}/>
          </div>
        </div>
      </div>;
    }
    export default App;

```

APP.TSX

```

import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
  let dashboardObj: DashboardLayoutComponent;
  let buttonObj: any;
  let resetPanels : any = [];
  const cellSpacing: number[] = [10, 10];
  let panels: any = [
    { 'sizeX': 2, 'sizeY': 2, 'row': 1, 'col': 0, content: '<div
class="content">0</div>' },
    { 'sizeX': 2, 'sizeY': 2, 'row': 2, 'col': 2, content: '<div
class="content">1</div>' },
    { 'sizeX': 2, 'sizeY': 2, 'row': 3, 'col': 4, content: '<div
class="content">2</div>' }
  ];
  // Button click action to change the floating options for DashboardLayout
  function onClick(args: any): void {
    if (buttonObj.value === "Disable Floating and Reset") {
      buttonObj.value = 'Enable Floating';
      dashboardObj.allowFloating = false;
      dashboardObj.panels = resetPanels;
    } else {
      buttonObj.value = 'Disable Floating and Reset';
      dashboardObj.allowFloating = true;
    }
  }
  // Using created event to initialize the panels
  function onCreate(args: any): void {
    resetPanels = dashboardObj.serialize();
    resetPanels[0].content = '<div class="content">0</div>';
    resetPanels[1].content = '<div class="content">1</div>';
    resetPanels[2].content = '<div class="content">2</div>';
  }
  return (
    <div>
      <div id='container'>
        <div className="inline" id="control">

```

```

        <DashboardLayoutComponent id='defaultLayout' ref={ s =>
        (dashboardObj = s as DashboardLayoutComponent)} cellSpacing={cellSpacing}
        created= { onCreate } panels={panels} allowFloating={false}
        cellAspectRatio={100/75} columns={6} />
      </div>
      <div className="inline" id="properties">
        <input type="button" ref={ s => buttonObj = s} className= "e-
        btn e-flat e-primary e-outline" name="floating" id="floating" value="Enable
        Floating" onClick={ onClick } />
      </div>
    </div>
  </div>
);
}
export default App;

```

INDEX.JSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

INDEX.TSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

You can refer to our [React Dashboard Layout](#) feature tour page for its groundbreaking feature representations. You can also explore our [React Dashboard Layout example](#) to know how to present and manipulate data.

Responsive adaptive in React Dashboard layout component

The control is provided with built-in responsive support, where panels within the layout get adjusted based on their parent element's dimensions to accommodate any resolution which relieves the burden of building responsive dashboards.

The dashboard layout is designed to automatically adapt with lower resolutions by transforming the entire layout into a stacked one, so that, the panels will be displayed in a vertical column. By default, whenever the screen resolution meets 600 px or lower resolutions this layout transformation occurs. This transformation can be modified for any user defined resolution by defining the `mediaQuery` property of the component.

The following sample demonstrates the usage of the `mediaQuery` property to turn out the layout into a stacked one in user defined resolution. Here, whenever, the window size reaches 700 px or lesser, the layout becomes a stacked layout.

APP.JSX

```

import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {

```

```

const cellSpacing = [20, 20];
let mediaQuery = 'max-width: 700px';
let panels = [
  { "sizeX": 1, "sizeY": 1, "row": 0, "col": 0, content: '<div
class="content">0</div>' },
  { "sizeX": 3, "sizeY": 2, "row": 0, "col": 1, content: '<div
class="content">1</div>' },
  { "sizeX": 1, "sizeY": 3, "row": 0, "col": 4, content: '<div
class="content">2</div>' },
  { "sizeX": 1, "sizeY": 1, "row": 1, "col": 0, content: '<div
class="content">3</div>' },
  { "sizeX": 2, "sizeY": 1, "row": 2, "col": 0, content: '<div
class="content">4</div>' },
  { "sizeX": 1, "sizeY": 1, "row": 2, "col": 2, content: '<div
class="content">5</div>' },
  { "sizeX": 1, "sizeY": 1, "row": 2, "col": 3, content: '<div
class="content">6</div>' }
];
return (<div>
  <div id='container'>
    <DashboardLayoutComponent id='defaultLayout' columns={5}
cellSpacing={cellSpacing} panels={panels} mediaQuery={mediaQuery}/>
  </div>
</div>);
}
export default App;

```

APP.TSX

```

import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
  const cellSpacing: number[] = [20, 20];
  let mediaQuery: string = 'max-width: 700px';
  let panels: any = [
    { "sizeX": 1, "sizeY": 1, "row": 0, "col": 0, content: '<div
class="content">0</div>' },
    { "sizeX": 3, "sizeY": 2, "row": 0, "col": 1, content: '<div
class="content">1</div>' },
    { "sizeX": 1, "sizeY": 3, "row": 0, "col": 4, content: '<div
class="content">2</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 1, "col": 0, content: '<div
class="content">3</div>' },
    { "sizeX": 2, "sizeY": 1, "row": 2, "col": 0, content: '<div
class="content">4</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 2, content: '<div
class="content">5</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 3, content: '<div
class="content">6</div>' }
  ];
  return (
    <div>
      <div id='container'>
        <DashboardLayoutComponent id='defaultLayout' columns={5}
cellSpacing={cellSpacing} panels={panels} mediaQuery={mediaQuery} />
      </div>
    </div>
  );
}

```

```

    </div>
  );
}
export default App;

```

INDEX.JSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

INDEX.TSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

You can refer to our [React Dashboard Layout](#) feature tour page for its groundbreaking feature representations. You can also explore our [React Dashboard Layout example](#) to know how to present and manipulate data.

State maintenance in React Dashboard layout component

The current layout structure of the Dashboard Layout component can be obtained and saved to construct another dashboard with same panel structure using the `serialize` public method of the component. This method returns the component's current panel setting which can be used to construct a dashboard with the same layout settings.

The following sample demonstrates how to save and restore the state of the panels using the `serialize` method. Click Save to store the panel's settings and click Restore to restore the previously saved panel settings.

APP.JSX

```

import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
  let dashboardObj;
  const cellSpacing = [20, 20];
  let panels = [
    { "sizeX": 1, "sizeY": 1, "row": 0, "col": 0, content: '<div class="content">0</div>' },
    { "sizeX": 3, "sizeY": 2, "row": 0, "col": 1, content: '<div class="content">1</div>' },
    { "sizeX": 1, "sizeY": 3, "row": 0, "col": 4, content: '<div class="content">2</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 1, "col": 0, content: '<div class="content">3</div>' },
    { "sizeX": 2, "sizeY": 1, "row": 2, "col": 0, content: '<div class="content">4</div>' },
    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 2, content: '<div class="content">5</div>' },
  ];

```



```

    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 3, content: '<div
class="content">6</div>' }
];
let restoreModel = [];
function onRestore(args) {
    dashboardObj.panels = restoreModel;
}
function onSave(args) {
    restoreModel = dashboardObj.serialize();
    restoreModel[0].content = '<div class="content">0</div>';
    restoreModel[1].content = '<div class="content">1</div>';
    restoreModel[2].content = '<div class="content">2</div>';
    restoreModel[3].content = '<div class="content">3</div>';
    restoreModel[4].content = '<div class="content">4</div>';
    restoreModel[5].content = '<div class="content">5</div>';
    restoreModel[6].content = '<div class="content">6</div>';
}
return (<div>
    <div id='container'>
        <div className="inline" id="control">
            <DashboardLayoutComponent id='defaultLayout' created={onSave}
ref={s => (dashboardObj = s)} cellSpacing={cellSpacing} panels={panels}
columns={5}/>
        </div>
        <div className="inline" id="properties">
            <button className="e-btn e-primary"
onClick={onSave}>Save</button>
            <button className="e-btn e-flat e-outline" id="Restore"
onClick={onRestore}>Restore</button>
        </div>
    </div>
</div>);
}
export default App;

```

APP.TSX

```

import { DashboardLayoutComponent } from '@syncfusion/ej2-react-layouts';
import * as React from 'react';
function App() {
    let dashboardObj: DashboardLayoutComponent;
    const cellSpacing: number[] = [20, 20];
    let panels: any = [
        { "sizeX": 1, "sizeY": 1, "row": 0, "col": 0, content: '<div
class="content">0</div>' },
        { "sizeX": 3, "sizeY": 2, "row": 0, "col": 1, content: '<div
class="content">1</div>' },
        { "sizeX": 1, "sizeY": 3, "row": 0, "col": 4, content: '<div
class="content">2</div>' },
        { "sizeX": 1, "sizeY": 1, "row": 1, "col": 0, content: '<div
class="content">3</div>' },
        { "sizeX": 2, "sizeY": 1, "row": 2, "col": 0, content: '<div
class="content">4</div>' },
        { "sizeX": 1, "sizeY": 1, "row": 2, "col": 2, content: '<div
class="content">5</div>' },
    ];
}

```

```

    { "sizeX": 1, "sizeY": 1, "row": 2, "col": 3, content: '<div
class="content">6</div>' }
  ];
  let restoreModel: any = [];
  function onRestore(args: any): void {
    dashboardObj.panels = restoreModel;
  }
  function onSave(args: any): void {
    restoreModel = dashboardObj.serialize();
    restoreModel[0].content = '<div class="content">0</div>';
    restoreModel[1].content = '<div class="content">1</div>';
    restoreModel[2].content = '<div class="content">2</div>';
    restoreModel[3].content = '<div class="content">3</div>';
    restoreModel[4].content = '<div class="content">4</div>';
    restoreModel[5].content = '<div class="content">5</div>';
    restoreModel[6].content = '<div class="content">6</div>';
  }
  return (
    <div>
      <div id='container'>
        <div className="inline" id="control">
          <DashboardLayoutComponent id='defaultLayout' created={ onSave }
ref={ s => (dashboardObj = s as DashboardLayoutComponent) }
cellSpacing={cellSpacing} panels={panels} columns={5} />
        </div>
        <div className="inline" id="properties">
          <button className="e-btn e-primary"
onClick={onSave}>Save</button>
          <button className="e-btn e-flat e-outline" id="Restore"
onClick={onRestore}>Restore</button>
        </div>
      </div>
    </div>
  );
}
export default App;

```

INDEX.JSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

INDEX.TSX

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
import App from './App';
ReactDOM.render(<App />, document.getElementById('root'));

```

You can refer to our [React Dashboard Layout](#) feature tour page for its groundbreaking feature representations. You can also explore our [React Dashboard Layout example](#) to know how to present and manipulate data.

Style in React Dashboard layout component

The following content provides the exact CSS structure that can be used to modify the control's appearance based on the user preference.

Customizing the dashboard layout panel header

Use the following CSS to customize the dashboard layout panel header.

```
`css
.e-dashboardlayout.e-control .e-panel .e-panel-container .e-panel-header {
color: #754131;
background-color: #c9e2f7;
text-align: center;
}
`
```

Customizing the dashboard layout panel content

Use the following CSS to customize the dashboard layout panel content.

```
`css
.e-dashboardlayout.e-control .e-panel .e-panel-container .e-panel-content {
background-color: #c9e2f7;
padding: 50px;
}
`
```

Customizing the dashboard layout panel resize icon

Use the following CSS to customize the dashboard layout resize icon.

```
`css
.e-dashboardlayout.e-control .e-panel .e-panel-container .e-resize.e-double{
color: #0378d5;
font-size: 30px;
height: 20px;
width: 20px;
}
`
```

Customizing the dashboard layout panel background

Use the following CSS to customize the dashboard layout panel background.

```
`css
.e-dashboardlayout.e-control.e-responsive {
background: #b3d3ed;
}
```

```
}
,
```

You can refer to our [React Dashboard Layout](#) feature tour page for its groundbreaking feature representations. You can also explore our [React Dashboard Layout example](#) to know how to present and manipulate data.

Accessibility in React Dashboard Layout component

The Dashboard Layout component followed the accessibility guidelines and standards, including [ADA, Section 508](#), [WCAG 2.2](#) standards, and [WCAG roles](#) that are commonly used to evaluate accessibility.

The accessibility compliance for the Dashboard Layout component is outlined below.

| Accessibility Criteria | Compatibility |

| -- | -- |

| [WCAG 2.2](#) Support | |

| [Section 508](#) Support | |

| Screen Reader Support | |

| Right-To-Left Support | |

| Color Contrast | |

| Mobile Device Support | |

| Keyboard Navigation Support | Not applicable |

| [Accessibility Checker](#) Validation | |

| [Axe-core](#) Accessibility Validation | |

<style>

.post .post-content img {

display: inline-block;

margin: 0.5em 0;

}

</style>

<div> - All features of the component meet the requirement.</div>

```
<div> - Some features of the component do not meet the requirement.</div>
```

```
<div> - The component does not meet the requirement.</div>
```

WAI-ARIA attributes

The Dashboard Layout component followed the [WAI-ARIA](#) patterns to meet the accessibility. The following ARIA attributes are used in the Dashboard Layout component:

| **Attributes** | **Purpose** |

| --- | --- |

| **role=list** | Indicates the role as a list for the Dashboard Layout element. |

| **role=listitem** | Indicates the role as a listitem for the Dashboard panels. |

| **role=presentation** | Indicates the role as a presentation for the table when the **showGridLines** property is enabled. |

| **aria-grabbed** | When the panel is chosen for dragging, the aria-grabbed attribute is set to "true". If it's set to "false", the element can be grabbed for drag-and-drop, but it won't be actively held. |

Keyboard interaction

Keyboard support is not applicable for the Dashboard Layout.

Ensuring accessibility

The Dashboard Layout component's accessibility levels are ensured through an [accessibility-checker](#) and [axe-core](#) software tools during automated testing.

The accessibility compliance of the Dashboard Layout component is shown in the following sample. Open the [sample](#) in a new window to evaluate the accessibility of the Dashboard Layout component with accessibility tools.

See also

- [Accessibility in Syncfusion React components](#)

DataManager

Getting started

Dependencies

Below is the list of minimum dependencies required to use the DataManager.

```
`javascript
```

```
|-- @syncfusion/ej2-data
```

```
|-- @syncfusion/ej2-base
```

```
|-- es6-promise (Required when window.Promise is not available)
```

```
`
```

@syncfusion/ej2-data requires the presence of a Promise feature in global environment. In the browser, window.Promise must be available.

Installation and configuration

To set-up a React application, choose any of the following ways. The best and easiest way is to use the [create-react-app](#). It sets up your development environment in JavaScript and improvise your application for production. Refer to the [installation instructions](#) of `create-react-app`.

```
`bash
npx create-react-app my-app
cd my-app
npm start
`
```

or

```
`bash
yarn create react-app my-app
cd my-app
yarn start
`
```

To set-up a React application in `TypeScript` environment, run the following command.

```
`bash
npx create-react-app my-app --template typescript
cd my-app
npm start
`
```

Besides using the [npm](#) package runner tool, also create an application from the `npm init`. To begin with the `npm init`, upgrade the `npm` version to `npm 6+`.

```
`bash
npm init react-app my-app
cd my-app
npm start
`
```

Install Syncfusion packages using below command.

```
`
npm install @syncfusion/ej2-data --save
`
```

Connection to a data source

The DataManager can act as gateway for both local and remote data source which will uses the query to interact with the data source.

Binding to JSON data

[DataManager](#) can be bound to local data source by assigning the array of JavaScript objects to the **json** property or simply passing them to the constructor while instantiating.

Add the CSS below to the **app/app.css** file to style the table, then import it in the **src/app.tsx** or **src/app.jsx** file.

```
`css
```

```
datatable {
```

```
border: solid 1px #e0e0e0;
```

```
border-collapse: collapse;
```

```
font-family: Roboto;
```

```
}
```

```
datatable td,
```

```
datatable th {
```

```
border: solid #e0e0e0;
```

```
border-width: 1px 0 0;
```

```
display: table-cell;
```

```
font-size: 14px;
```

```
line-height: 20px;
```

```
overflow: hidden;
```

```
padding: 8px 21px;
```

```
vertical-align: middle;
```

```
white-space: nowrap;
```

```
width: auto;
```

```
}
```

```
,
```

APP.JSX

```
import { DataManager, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { Row } from './rowTemplate';
export default class App extends React.Component {
  result = new DataManager(data).executeLocal(new Query().take(8));
  items = this.result.map((row, index) => (
    <Row key={index} {...row} />
  ));
  render() {
    return (
      <table id='datatable' className='e-table'>
        <thead>
          <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
```

```

        </thead>
        <tbody>{this.items}</tbody>
    </table>;
    }
}

```

APP.TSX

```

import { DataManager, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { IOrders } from './orders';
import { Row } from './rowTemplate';
export default class App extends React.Component<{}, {}>{
    public result: IOrders[] = new DataManager(data).executeLocal(new
    Query().take(8)) as IOrders[];
    public items: React.ReactNode[] = this.result.map((row: IOrders, index)
=> (
        <Row key={index} {...row} />
    ));
    public render() {
        return <table id='datatable' className='e-table'>
            <thead>
                <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
            </thead>
            <tbody>{this.items}</tbody>
        </table>;
    }
}

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```

export interface IOrders {
    OrderID: number;
    EmployeeID: number;
    CustomerID: string;
    Order_Details: object[];
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
export class Row extends React.Component {
    render() {
        const item = this.props;
        return <tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>;
    }
}

```



```
}
}
```

ROWTEMPLATE.TSX

```
import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
    public render() {
        const item: IOrders = this.props as IOrders;
        return (<tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>)
    }
}
```

DATASOURCE.JSX

```
export let data = [
    {
        OrderID: 10248, CustomerID: 'VINET', EmployeeID: 5, OrderDate: new
        Date(8364186e5),
        ShipName: 'Vins et alcools Chevalier', ShipCity: 'Reims',
        ShipAddress: '59 rue de l Abbaye',
        ShipRegion: 'CJ', ShipPostalCode: '51100', ShipCountry: 'France',
        Freight: 32.38, Verified: !0
    },
    {
        OrderID: 10249, CustomerID: 'TOMSP', EmployeeID: 6, OrderDate: new
        Date(836505e6),
        ShipName: 'Toms Spezialitäten', ShipCity: 'Münster', ShipAddress:
        'Luisenstr. 48',
        ShipRegion: 'CJ', ShipPostalCode: '44087', ShipCountry: 'Germany',
        Freight: 11.61, Verified: !1
    },
    {
        OrderID: 10250, CustomerID: 'HANAR', EmployeeID: 4, OrderDate: new
        Date(8367642e5),
        ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
        'Rua do Paço, 67',
        ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
        Freight: 65.83, Verified: !0
    },
    {
        OrderID: 10251, CustomerID: 'VICTE', EmployeeID: 3, OrderDate: new
        Date(8367642e5),
        ShipName: 'Victuailles en stock', ShipCity: 'Lyon', ShipAddress: '2,
        rue du Commerce',
        ShipRegion: 'CJ', ShipPostalCode: '69004', ShipCountry: 'France',
        Freight: 41.34, Verified: !0
    },
    {
        OrderID: 10252, CustomerID: 'SUPRD', EmployeeID: 4, OrderDate: new
        Date(8368506e5),
```

```

        ShipName: 'Suprêmes délices', ShipCity: 'Charleroi', ShipAddress:
        'Boulevard Tirou, 255',
        ShipRegion: 'CJ', ShipPostalCode: 'B-6000', ShipCountry: 'Belgium',
        Freight: 51.3, Verified: !0
    },
    {
        OrderID: 10253, CustomerID: 'HANAR', EmployeeID: 3, OrderDate: new
        Date(836937e6),
        ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
        'Rua do Paço, 67',
        ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
        Freight: 58.17, Verified: !0
    },
    {
        OrderID: 10254, CustomerID: 'CHOPS', EmployeeID: 5, OrderDate: new
        Date(8370234e5),
        ShipName: 'Chop-suey Chinese', ShipCity: 'Bern', ShipAddress:
        'Hauptstr. 31',
        ShipRegion: 'CJ', ShipPostalCode: '3012', ShipCountry: 'Switzerland',
        Freight: 22.98, Verified: !1
    },
    {
        OrderID: 10255, CustomerID: 'RICSU', EmployeeID: 9, OrderDate: new
        Date(8371098e5),
        ShipName: 'Richter Supermarkt', ShipCity: 'Genève', ShipAddress:
        'Starenweg 5',
        ShipRegion: 'CJ', ShipPostalCode: '1204', ShipCountry: 'Switzerland',
        Freight: 148.33, Verified: !0
    },
    {
        OrderID: 10256, CustomerID: 'WELLI', EmployeeID: 3, OrderDate: new
        Date(837369e6),
        ShipName: 'Wellington Importadora', ShipCity: 'Resende', ShipAddress:
        'Rua do Mercado, 12',
        ShipRegion: 'SP', ShipPostalCode: '08737-363', ShipCountry: 'Brazil',
        Freight: 13.97, Verified: !1
    },
    {
        OrderID: 10257, CustomerID: 'HILAA', EmployeeID: 4, OrderDate: new
        Date(8374554e5),
        ShipName: 'HILARION-Abastos', ShipCity: 'San Cristóbal', ShipAddress:
        'Carrera 22 con Ave. Carlos Soublette #8-35',
        ShipRegion: 'Táchira', ShipPostalCode: '5022', ShipCountry:
        'Venezuela', Freight: 81.91, Verified: !0
    },
    {
        OrderID: 10258, CustomerID: 'ERNSH', EmployeeID: 1, OrderDate: new
        Date(8375418e5),
        ShipName: 'Ernst Handel', ShipCity: 'Graz', ShipAddress: 'Kirchgasse
        6',
        ShipRegion: 'CJ', ShipPostalCode: '8010', ShipCountry: 'Austria',
        Freight: 140.51, Verified: !0
    },
    {
        OrderID: 10259, CustomerID: 'CENTC', EmployeeID: 4, OrderDate: new
        Date(8376282e5),

```

```

        ShipName: 'Centro comercial Moctezuma', ShipCity: 'México D.F.',
        ShipAddress: 'Sierras de Granada 9993',
        ShipRegion: 'CJ', ShipPostalCode: '05022', ShipCountry: 'Mexico',
        Freight: 3.25, Verified: !1
    },
    {
        OrderID: 10260, CustomerID: 'OTTIK', EmployeeID: 4, OrderDate: new
        Date(8377146e5),
        ShipName: 'Ottilies Käseladen', ShipCity: 'Köln', ShipAddress:
        'Mehrheimerstr. 369',
        ShipRegion: 'CJ', ShipPostalCode: '50739', ShipCountry: 'Germany',
        Freight: 55.09, Verified: !0
    },
    {
        OrderID: 10261, CustomerID: 'QUEDE', EmployeeID: 4, OrderDate: new
        Date(8377146e5),
        ShipName: 'Que Delícia', ShipCity: 'Rio de Janeiro', ShipAddress:
        'Rua da Panificadora, 12',
        ShipRegion: 'RJ', ShipPostalCode: '02389-673', ShipCountry: 'Brazil',
        Freight: 3.05, Verified: !1
    },
    {
        OrderID: 10262, CustomerID: 'RATTC', EmployeeID: 8, OrderDate: new
        Date(8379738e5),
        ShipName: 'Rattlesnake Canyon Grocery', ShipCity: 'Albuquerque',
        ShipAddress: '2817 Milton Dr.',
        ShipRegion: 'NM', ShipPostalCode: '87110', ShipCountry: 'USA',
        Freight: 48.29, Verified: !0
    }
    ]];

```

DATASOURCE.TSX

```

export let data: Object[] = [
    {
        OrderID: 10248, CustomerID: 'VINET', EmployeeID: 5, OrderDate: new
        Date(8364186e5),
        ShipName: 'Vins et alcools Chevalier', ShipCity: 'Reims',
        ShipAddress: '59 rue de l Abbaye',
        ShipRegion: 'CJ', ShipPostalCode: '51100', ShipCountry: 'France',
        Freight: 32.38, Verified: !0
    },
    {
        OrderID: 10249, CustomerID: 'TOMSP', EmployeeID: 6, OrderDate: new
        Date(836505e6),
        ShipName: 'Toms Spezialitäten', ShipCity: 'Münster', ShipAddress:
        'Luisenstr. 48',
        ShipRegion: 'CJ', ShipPostalCode: '44087', ShipCountry: 'Germany',
        Freight: 11.61, Verified: !1
    },
    {
        OrderID: 10250, CustomerID: 'HANAR', EmployeeID: 4, OrderDate: new
        Date(8367642e5),
        ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
        'Rua do Paço, 67',
        ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
        Freight: 65.83, Verified: !0
    }
];

```

```

    },
    {
        OrderID: 10251, CustomerID: 'VICTE', EmployeeID: 3, OrderDate: new
Date(8367642e5),
        ShipName: 'Victuailles en stock', ShipCity: 'Lyon', ShipAddress: '2,
rue du Commerce',
        ShipRegion: 'CJ', ShipPostalCode: '69004', ShipCountry: 'France',
Freight: 41.34, Verified: !0
    },
    {
        OrderID: 10252, CustomerID: 'SUPRD', EmployeeID: 4, OrderDate: new
Date(8368506e5),
        ShipName: 'Suprêmes délices', ShipCity: 'Charleroi', ShipAddress:
'Boulevard Tirou, 255',
        ShipRegion: 'CJ', ShipPostalCode: 'B-6000', ShipCountry: 'Belgium',
Freight: 51.3, Verified: !0
    },
    {
        OrderID: 10253, CustomerID: 'HANAR', EmployeeID: 3, OrderDate: new
Date(836937e6),
        ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
'Rua do Paço, 67',
        ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
Freight: 58.17, Verified: !0
    },
    {
        OrderID: 10254, CustomerID: 'CHOPS', EmployeeID: 5, OrderDate: new
Date(8370234e5),
        ShipName: 'Chop-suey Chinese', ShipCity: 'Bern', ShipAddress:
'Hauptstr. 31',
        ShipRegion: 'CJ', ShipPostalCode: '3012', ShipCountry: 'Switzerland',
Freight: 22.98, Verified: !1
    },
    {
        OrderID: 10255, CustomerID: 'RICSU', EmployeeID: 9, OrderDate: new
Date(8371098e5),
        ShipName: 'Richter Supermarkt', ShipCity: 'Genève', ShipAddress:
'Starenweg 5',
        ShipRegion: 'CJ', ShipPostalCode: '1204', ShipCountry: 'Switzerland',
Freight: 148.33, Verified: !0
    },
    {
        OrderID: 10256, CustomerID: 'WELLI', EmployeeID: 3, OrderDate: new
Date(837369e6),
        ShipName: 'Wellington Importadora', ShipCity: 'Resende', ShipAddress:
'Rua do Mercado, 12',
        ShipRegion: 'SP', ShipPostalCode: '08737-363', ShipCountry: 'Brazil',
Freight: 13.97, Verified: !1
    },
    {
        OrderID: 10257, CustomerID: 'HILAA', EmployeeID: 4, OrderDate: new
Date(8374554e5),
        ShipName: 'HILARION-Abastos', ShipCity: 'San Cristóbal', ShipAddress:
'Carrera 22 con Ave. Carlos Soublette #8-35',
        ShipRegion: 'Táchira', ShipPostalCode: '5022', ShipCountry:
'Venezuela', Freight: 81.91, Verified: !0
    },

```

```

{
    OrderID: 10258, CustomerID: 'ERNSH', EmployeeID: 1, OrderDate: new
Date(8375418e5),
    ShipName: 'Ernst Handel', ShipCity: 'Graz', ShipAddress: 'Kirchgasse
6',
    ShipRegion: 'CJ', ShipPostalCode: '8010', ShipCountry: 'Austria',
Freight: 140.51, Verified: !0
},
{
    OrderID: 10259, CustomerID: 'CENTC', EmployeeID: 4, OrderDate: new
Date(8376282e5),
    ShipName: 'Centro comercial Moctezuma', ShipCity: 'México D.F.',
ShipAddress: 'Sierras de Granada 9993',
    ShipRegion: 'CJ', ShipPostalCode: '05022', ShipCountry: 'Mexico',
Freight: 3.25, Verified: !1
},
{
    OrderID: 10260, CustomerID: 'OTTIK', EmployeeID: 4, OrderDate: new
Date(8377146e5),
    ShipName: 'Ottilies Käseladen', ShipCity: 'Köln', ShipAddress:
'Mehrheimerstr. 369',
    ShipRegion: 'CJ', ShipPostalCode: '50739', ShipCountry: 'Germany',
Freight: 55.09, Verified: !0
},
{
    OrderID: 10261, CustomerID: 'QUEDE', EmployeeID: 4, OrderDate: new
Date(8377146e5),
    ShipName: 'Que Delícia', ShipCity: 'Rio de Janeiro', ShipAddress:
'Rua da Panificadora, 12',
    ShipRegion: 'RJ', ShipPostalCode: '02389-673', ShipCountry: 'Brazil',
Freight: 3.05, Verified: !1
},
{
    OrderID: 10262, CustomerID: 'RATTC', EmployeeID: 8, OrderDate: new
Date(8379738e5),
    ShipName: 'Rattlesnake Canyon Grocery', ShipCity: 'Albuquerque',
ShipAddress: '2817 Milton Dr.',
    ShipRegion: 'NM', ShipPostalCode: '87110', ShipCountry: 'USA',
Freight: 48.29, Verified: !0
}]]];

```

APP.CSS

```

#datatable {
    border: solid 1px #e0e0e0;
    border-collapse: collapse;
    font-family: Roboto;
}
#datatable td,
#datatable th {
    border: solid #e0e0e0;
    border-width: 1px 0 0;
    display: table-cell;
    font-size: 14px;
    line-height: 20px;
    overflow: hidden;
}

```

```
padding: 8px 21px;
vertical-align: middle;
white-space: nowrap;
width: auto;
}
```

Binding to ODataV4

[DataManager](#) can be bound to remote data source by assigning service end point URL to the `url` property. Now all `DataManager` operations will address the provided service end point.

APP.JSX

```
import { getValue } from '@syncfusion/ej2-base';
import { DataManager, Query, ODataV4Adaptor } from '@syncfusion/ej2-data';
import * as React from 'react';
import { Row } from './rowTemplate';
const SERVICE_URI =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = { items: [] };
  }
  componentDidMount() {
    new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor()
  }).executeQuery(new Query().take(8))
    .then((e) => {
      const res = e.result.map((row) => <Row key={row.OrderID} {...row}
/>);
      this.setState({
        items: res,
      });
    });
  }
  render() {
    return (
      <table id='datatable' className='e-table'>
        <thead>
          <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
        </thead>
        <tbody>{getValue('items', this.state)}</tbody>
      </table>);
  }
}
```

APP.TSX

```
import { getValue } from '@syncfusion/ej2-base';
import { DataManager, Query, ReturnOption, ODataV4Adaptor } from
 '@syncfusion/ej2-data';
import * as React from 'react';
import { IOOrders } from './orders';
import { Row } from './rowTemplate';
const SERVICE_URI: string =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
```

```

export default class App extends React.Component<{}, {}>{
  constructor(props: object) {
    super(props);
    this.state = { items: [] };
  }
  componentDidMount() {
    new DataManager({ url: SERVICE_URI ,adaptor: new ODataV4Adaptor()
  }).executeQuery(new Query().take(8))
    .then((e: ReturnOption) => {
      const res = (e.result as IOOrders[]).map((row: IOOrders, index:
number) => (
        <Row key={row.OrderID} {...row} />
      ));
      this.setState({
        items: res
      });
    });
  }
  public render() {
    return <table id='datatable' className='e-table'>
      <thead>
        <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
      </thead>
      <tbody>{ getValue('items', this.state) }</tbody>
    </table>
  }
}

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```

export interface IOOrders {
  OrderID: number;
  EmployeeID: number;
  CustomerID: string;
  // Order_Details: object[];
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
export class Row extends React.Component {
  render() {
    const item = this.props;
    return <tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>;
  }
}

```

ROWTEMPLATE.TSX

```
import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
    public render() {
        const item: IOrders = this.props as IOrders;
        return (<tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>)
    }
}
```

DATASOURCE.JSX

```
export let data = [
    {
        OrderID: 10248, CustomerID: 'VINET', EmployeeID: 5, OrderDate: new
        Date(8364186e5),
        ShipName: 'Vins et alcools Chevalier', ShipCity: 'Reims',
        ShipAddress: '59 rue de l Abbaye',
        ShipRegion: 'CJ', ShipPostalCode: '51100', ShipCountry: 'France',
        Freight: 32.38, Verified: !0
    },
    {
        OrderID: 10249, CustomerID: 'TOMSP', EmployeeID: 6, OrderDate: new
        Date(836505e6),
        ShipName: 'Toms Spezialitäten', ShipCity: 'Münster', ShipAddress:
        'Luisenstr. 48',
        ShipRegion: 'CJ', ShipPostalCode: '44087', ShipCountry: 'Germany',
        Freight: 11.61, Verified: !1
    },
    {
        OrderID: 10250, CustomerID: 'HANAR', EmployeeID: 4, OrderDate: new
        Date(8367642e5),
        ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
        'Rua do Paço, 67',
        ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
        Freight: 65.83, Verified: !0
    },
    {
        OrderID: 10251, CustomerID: 'VICTE', EmployeeID: 3, OrderDate: new
        Date(8367642e5),
        ShipName: 'Victuailles en stock', ShipCity: 'Lyon', ShipAddress: '2,
        rue du Commerce',
        ShipRegion: 'CJ', ShipPostalCode: '69004', ShipCountry: 'France',
        Freight: 41.34, Verified: !0
    },
    {
        OrderID: 10252, CustomerID: 'SUPRD', EmployeeID: 4, OrderDate: new
        Date(8368506e5),
        ShipName: 'Suprêmes délices', ShipCity: 'Charleroi', ShipAddress:
        'Boulevard Tirou, 255',
```



```

        ShipRegion: 'CJ', ShipPostalCode: 'B-6000', ShipCountry: 'Belgium',
        Freight: 51.3, Verified: !0
    },
    {
        OrderID: 10253, CustomerID: 'HANAR', EmployeeID: 3, OrderDate: new
        Date(836937e6),
        ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
        'Rua do Paço, 67',
        ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
        Freight: 58.17, Verified: !0
    },
    {
        OrderID: 10254, CustomerID: 'CHOPS', EmployeeID: 5, OrderDate: new
        Date(8370234e5),
        ShipName: 'Chop-suey Chinese', ShipCity: 'Bern', ShipAddress:
        'Hauptstr. 31',
        ShipRegion: 'CJ', ShipPostalCode: '3012', ShipCountry: 'Switzerland',
        Freight: 22.98, Verified: !1
    },
    {
        OrderID: 10255, CustomerID: 'RICSU', EmployeeID: 9, OrderDate: new
        Date(8371098e5),
        ShipName: 'Richter Supermarkt', ShipCity: 'Genève', ShipAddress:
        'Starenweg 5',
        ShipRegion: 'CJ', ShipPostalCode: '1204', ShipCountry: 'Switzerland',
        Freight: 148.33, Verified: !0
    },
    {
        OrderID: 10256, CustomerID: 'WELLI', EmployeeID: 3, OrderDate: new
        Date(837369e6),
        ShipName: 'Wellington Importadora', ShipCity: 'Resende', ShipAddress:
        'Rua do Mercado, 12',
        ShipRegion: 'SP', ShipPostalCode: '08737-363', ShipCountry: 'Brazil',
        Freight: 13.97, Verified: !1
    },
    {
        OrderID: 10257, CustomerID: 'HILAA', EmployeeID: 4, OrderDate: new
        Date(8374554e5),
        ShipName: 'HILARION-Abastos', ShipCity: 'San Cristóbal', ShipAddress:
        'Carrera 22 con Ave. Carlos Soublette #8-35',
        ShipRegion: 'Táchira', ShipPostalCode: '5022', ShipCountry:
        'Venezuela', Freight: 81.91, Verified: !0
    },
    {
        OrderID: 10258, CustomerID: 'ERNSH', EmployeeID: 1, OrderDate: new
        Date(8375418e5),
        ShipName: 'Ernst Handel', ShipCity: 'Graz', ShipAddress: 'Kirchgasse
        6',
        ShipRegion: 'CJ', ShipPostalCode: '8010', ShipCountry: 'Austria',
        Freight: 140.51, Verified: !0
    },
    {
        OrderID: 10259, CustomerID: 'CENTC', EmployeeID: 4, OrderDate: new
        Date(8376282e5),
        ShipName: 'Centro comercial Moctezuma', ShipCity: 'México D.F.',
        ShipAddress: 'Sierras de Granada 9993',

```

```

        ShipRegion: 'CJ', ShipPostalCode: '05022', ShipCountry: 'Mexico',
        Freight: 3.25, Verified: !1
    },
    {
        OrderID: 10260, CustomerID: 'OTTIK', EmployeeID: 4, OrderDate: new
        Date(8377146e5),
        ShipName: 'Ottilies Käseladen', ShipCity: 'Köln', ShipAddress:
        'Mehrheimerstr. 369',
        ShipRegion: 'CJ', ShipPostalCode: '50739', ShipCountry: 'Germany',
        Freight: 55.09, Verified: !0
    },
    {
        OrderID: 10261, CustomerID: 'QUEDE', EmployeeID: 4, OrderDate: new
        Date(8377146e5),
        ShipName: 'Que Delícia', ShipCity: 'Rio de Janeiro', ShipAddress:
        'Rua da Panificadora, 12',
        ShipRegion: 'RJ', ShipPostalCode: '02389-673', ShipCountry: 'Brazil',
        Freight: 3.05, Verified: !1
    },
    {
        OrderID: 10262, CustomerID: 'RATTC', EmployeeID: 8, OrderDate: new
        Date(8379738e5),
        ShipName: 'Rattlesnake Canyon Grocery', ShipCity: 'Albuquerque',
        ShipAddress: '2817 Milton Dr.',
        ShipRegion: 'NM', ShipPostalCode: '87110', ShipCountry: 'USA',
        Freight: 48.29, Verified: !0
    }
    ]];

```

DATASOURCE.TSX

```

export let data: Object[] = [
    {
        OrderID: 10248, CustomerID: 'VINET', EmployeeID: 5, OrderDate: new
        Date(8364186e5),
        ShipName: 'Vins et alcools Chevalier', ShipCity: 'Reims',
        ShipAddress: '59 rue de l Abbaye',
        ShipRegion: 'CJ', ShipPostalCode: '51100', ShipCountry: 'France',
        Freight: 32.38, Verified: !0
    },
    {
        OrderID: 10249, CustomerID: 'TOMSP', EmployeeID: 6, OrderDate: new
        Date(836505e6),
        ShipName: 'Toms Spezialitäten', ShipCity: 'Münster', ShipAddress:
        'Luisenstr. 48',
        ShipRegion: 'CJ', ShipPostalCode: '44087', ShipCountry: 'Germany',
        Freight: 11.61, Verified: !1
    },
    {
        OrderID: 10250, CustomerID: 'HANAR', EmployeeID: 4, OrderDate: new
        Date(8367642e5),
        ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
        'Rua do Paço, 67',
        ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
        Freight: 65.83, Verified: !0
    },
    {

```

```

    OrderID: 10251, CustomerID: 'VICTE', EmployeeID: 3, OrderDate: new
Date(8367642e5),
    ShipName: 'Victuailles en stock', ShipCity: 'Lyon', ShipAddress: '2,
rue du Commerce',
    ShipRegion: 'CJ', ShipPostalCode: '69004', ShipCountry: 'France',
Freight: 41.34, Verified: !0
    },
    {
        OrderID: 10252, CustomerID: 'SUPRD', EmployeeID: 4, OrderDate: new
Date(8368506e5),
        ShipName: 'Suprêmes délices', ShipCity: 'Charleroi', ShipAddress:
'Boulevard Tirou, 255',
        ShipRegion: 'CJ', ShipPostalCode: 'B-6000', ShipCountry: 'Belgium',
Freight: 51.3, Verified: !0
    },
    {
        OrderID: 10253, CustomerID: 'HANAR', EmployeeID: 3, OrderDate: new
Date(836937e6),
        ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
'Rua do Paço, 67',
        ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
Freight: 58.17, Verified: !0
    },
    {
        OrderID: 10254, CustomerID: 'CHOPS', EmployeeID: 5, OrderDate: new
Date(8370234e5),
        ShipName: 'Chop-suey Chinese', ShipCity: 'Bern', ShipAddress:
'Hauptstr. 31',
        ShipRegion: 'CJ', ShipPostalCode: '3012', ShipCountry: 'Switzerland',
Freight: 22.98, Verified: !1
    },
    {
        OrderID: 10255, CustomerID: 'RICSU', EmployeeID: 9, OrderDate: new
Date(8371098e5),
        ShipName: 'Richter Supermarkt', ShipCity: 'Genève', ShipAddress:
'Starenweg 5',
        ShipRegion: 'CJ', ShipPostalCode: '1204', ShipCountry: 'Switzerland',
Freight: 148.33, Verified: !0
    },
    {
        OrderID: 10256, CustomerID: 'WELLI', EmployeeID: 3, OrderDate: new
Date(837369e6),
        ShipName: 'Wellington Importadora', ShipCity: 'Resende', ShipAddress:
'Rua do Mercado, 12',
        ShipRegion: 'SP', ShipPostalCode: '08737-363', ShipCountry: 'Brazil',
Freight: 13.97, Verified: !1
    },
    {
        OrderID: 10257, CustomerID: 'HILAA', EmployeeID: 4, OrderDate: new
Date(8374554e5),
        ShipName: 'HILARION-Abastos', ShipCity: 'San Cristóbal', ShipAddress:
'Carrera 22 con Ave. Carlos Soublette #8-35',
        ShipRegion: 'Táchira', ShipPostalCode: '5022', ShipCountry:
'Venezuela', Freight: 81.91, Verified: !0
    },
    {

```

```

    OrderID: 10258, CustomerID: 'ERNSH', EmployeeID: 1, OrderDate: new
Date(8375418e5),
    ShipName: 'Ernst Handel', ShipCity: 'Graz', ShipAddress: 'Kirchgasse
6',
    ShipRegion: 'CJ', ShipPostalCode: '8010', ShipCountry: 'Austria',
Freight: 140.51, Verified: !0
    },
    {
        OrderID: 10259, CustomerID: 'CENTC', EmployeeID: 4, OrderDate: new
Date(8376282e5),
        ShipName: 'Centro comercial Moctezuma', ShipCity: 'México D.F.',
ShipAddress: 'Sierras de Granada 9993',
        ShipRegion: 'CJ', ShipPostalCode: '05022', ShipCountry: 'Mexico',
Freight: 3.25, Verified: !1
    },
    {
        OrderID: 10260, CustomerID: 'OTTIK', EmployeeID: 4, OrderDate: new
Date(8377146e5),
        ShipName: 'Ottilies Käseladen', ShipCity: 'Köln', ShipAddress:
'Mehrheimerstr. 369',
        ShipRegion: 'CJ', ShipPostalCode: '50739', ShipCountry: 'Germany',
Freight: 55.09, Verified: !0
    },
    {
        OrderID: 10261, CustomerID: 'QUEDE', EmployeeID: 4, OrderDate: new
Date(8377146e5),
        ShipName: 'Que Delícia', ShipCity: 'Rio de Janeiro', ShipAddress:
'Rua da Panificadora, 12',
        ShipRegion: 'RJ', ShipPostalCode: '02389-673', ShipCountry: 'Brazil',
Freight: 3.05, Verified: !1
    },
    {
        OrderID: 10262, CustomerID: 'RATTC', EmployeeID: 8, OrderDate: new
Date(8379738e5),
        ShipName: 'Rattlesnake Canyon Grocery', ShipCity: 'Albuquerque',
ShipAddress: '2817 Milton Dr.',
        ShipRegion: 'NM', ShipPostalCode: '87110', ShipCountry: 'USA',
Freight: 48.29, Verified: !0
    }
    ]];

```

APP.CSS

```

#datatable {
    border: solid 1px #e0e0e0;
    border-collapse: collapse;
    font-family: Roboto;
}
#datatable td,
#datatable th {
    border: solid #e0e0e0;
    border-width: 1px 0 0;
    display: table-cell;
    font-size: 14px;
    line-height: 20px;
    overflow: hidden;
    padding: 8px 21px;
}

```

```

vertical-align: middle;
white-space: nowrap;
width: auto;
}

```

Filter

The data filtering is a trivial operation which will let us to get reduced view of data based on filter criteria. The filter expression can be built easily using [where](#) method of [Query](#) class.

APP.JSX

```

import { DataManager, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { Row } from './rowTemplate';
export default class App extends React.Component {
  result = new DataManager(data).executeLocal(new Query()
    .where('EmployeeID', 'equal', 3));
  items = this.result.map((row, index) => (
    <Row key={index} {...row} />
  ));
  render() {
    return (
      <table id='datatable' className='e-table'>
        <thead>
          <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
        </thead>
        <tbody>{this.items}</tbody>
      </table>);
  }
}

```

APP.TSX

```

import { DataManager, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { IOrders } from './orders';
import { Row } from './rowTemplate';
export default class App extends React.Component<{}, {}>{
  public result: IOrders[] = new DataManager(data).executeLocal(new Query()
    .where('EmployeeID', 'equal', 3)) as IOrders[];
  public items: React.ReactNode[] = this.result.map((row: IOrders, index)
=> (
    <Row key={index} {...row} />
  ));
  public render() {
    return (
      <table id='datatable' className='e-table'>
        <thead>
          <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
        </thead>
        <tbody>{this.items}</tbody>
      </table>);
  }
}

```

```
}

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```
export interface IOrders {
  OrderID: number;
  EmployeeID: number;
  CustomerID: string;
  Order_Details: object[];
}
```

ROWTEMPLATE.JSX

```
import * as React from 'react';
export class Row extends React.Component {
  render() {
    const item = this.props;
    return <tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>;
  }
}
```

ROWTEMPLATE.TSX

```
import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
  public render() {
    const item: IOrders = this.props as IOrders;
    return <tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>
  }
}
```

APP.CSS

```
#datatable {
  border: solid 1px #e0e0e0;
  border-collapse: collapse;
  font-family: Roboto;
}
#datatable td,
#datatable th {
  border: solid #e0e0e0;
```

```

border-width: 1px 0 0;
display: table-cell;
font-size: 14px;
line-height: 20px;
overflow: hidden;
padding: 8px 21px;
vertical-align: middle;
white-space: nowrap;
width: auto;
}

```

DATASOURCE.JSX

```

export let data = [
  {
    OrderID: 10248, CustomerID: 'VINET', EmployeeID: 5, OrderDate: new
    Date(8364186e5),
    ShipName: 'Vins et alcools Chevalier', ShipCity: 'Reims',
    ShipAddress: '59 rue de l Abbaye',
    ShipRegion: 'CJ', ShipPostalCode: '51100', ShipCountry: 'France',
    Freight: 32.38, Verified: !0
  },
  {
    OrderID: 10249, CustomerID: 'TOMSP', EmployeeID: 6, OrderDate: new
    Date(836505e6),
    ShipName: 'Toms Spezialitäten', ShipCity: 'Münster', ShipAddress:
    'Luisenstr. 48',
    ShipRegion: 'CJ', ShipPostalCode: '44087', ShipCountry: 'Germany',
    Freight: 11.61, Verified: !1
  },
  {
    OrderID: 10250, CustomerID: 'HANAR', EmployeeID: 4, OrderDate: new
    Date(8367642e5),
    ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
    'Rua do Paço, 67',
    ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
    Freight: 65.83, Verified: !0
  },
  {
    OrderID: 10251, CustomerID: 'VICTE', EmployeeID: 3, OrderDate: new
    Date(8367642e5),
    ShipName: 'Victuailles en stock', ShipCity: 'Lyon', ShipAddress: '2,
    rue du Commerce',
    ShipRegion: 'CJ', ShipPostalCode: '69004', ShipCountry: 'France',
    Freight: 41.34, Verified: !0
  },
  {
    OrderID: 10252, CustomerID: 'SUPRD', EmployeeID: 4, OrderDate: new
    Date(8368506e5),
    ShipName: 'Suprêmes délices', ShipCity: 'Charleroi', ShipAddress:
    'Boulevard Tirou, 255',
    ShipRegion: 'CJ', ShipPostalCode: 'B-6000', ShipCountry: 'Belgium',
    Freight: 51.3, Verified: !0
  },
  {

```

```

    OrderID: 10253, CustomerID: 'HANAR', EmployeeID: 3, OrderDate: new
Date(836937e6),
    ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
'Rua do Paço, 67',
    ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
Freight: 58.17, Verified: !0
    },
    {
        OrderID: 10254, CustomerID: 'CHOPS', EmployeeID: 5, OrderDate: new
Date(8370234e5),
        ShipName: 'Chop-suey Chinese', ShipCity: 'Bern', ShipAddress:
'Hauptstr. 31',
        ShipRegion: 'CJ', ShipPostalCode: '3012', ShipCountry: 'Switzerland',
Freight: 22.98, Verified: !1
        },
        {
            OrderID: 10255, CustomerID: 'RICSU', EmployeeID: 9, OrderDate: new
Date(8371098e5),
            ShipName: 'Richter Supermarkt', ShipCity: 'Genève', ShipAddress:
'Starenweg 5',
            ShipRegion: 'CJ', ShipPostalCode: '1204', ShipCountry: 'Switzerland',
Freight: 148.33, Verified: !0
            },
            {
                OrderID: 10256, CustomerID: 'WELLI', EmployeeID: 3, OrderDate: new
Date(837369e6),
                ShipName: 'Wellington Importadora', ShipCity: 'Resende', ShipAddress:
'Rua do Mercado, 12',
                ShipRegion: 'SP', ShipPostalCode: '08737-363', ShipCountry: 'Brazil',
Freight: 13.97, Verified: !1
                },
                {
                    OrderID: 10257, CustomerID: 'HILAA', EmployeeID: 4, OrderDate: new
Date(8374554e5),
                    ShipName: 'HILARION-Abastos', ShipCity: 'San Cristóbal', ShipAddress:
'Carrera 22 con Ave. Carlos Soublette #8-35',
                    ShipRegion: 'Táchira', ShipPostalCode: '5022', ShipCountry:
'Venezuela', Freight: 81.91, Verified: !0
                    },
                    {
                        OrderID: 10258, CustomerID: 'ERNSH', EmployeeID: 1, OrderDate: new
Date(8375418e5),
                        ShipName: 'Ernst Handel', ShipCity: 'Graz', ShipAddress: 'Kirchgasse
6',
                        ShipRegion: 'CJ', ShipPostalCode: '8010', ShipCountry: 'Austria',
Freight: 140.51, Verified: !0
                        },
                        {
                            OrderID: 10259, CustomerID: 'CENTC', EmployeeID: 4, OrderDate: new
Date(8376282e5),
                            ShipName: 'Centro comercial Moctezuma', ShipCity: 'México D.F.',
ShipAddress: 'Sierras de Granada 9993',
                            ShipRegion: 'CJ', ShipPostalCode: '05022', ShipCountry: 'Mexico',
Freight: 3.25, Verified: !1
                            },
                            {

```



```

    OrderID: 10260, CustomerID: 'OTTIK', EmployeeID: 4, OrderDate: new
Date(8377146e5),
    ShipName: 'Ottilies Käseladen', ShipCity: 'Köln', ShipAddress:
'Mehrheimerstr. 369',
    ShipRegion: 'CJ', ShipPostalCode: '50739', ShipCountry: 'Germany',
Freight: 55.09, Verified: !0
    },
    {
        OrderID: 10261, CustomerID: 'QUEDE', EmployeeID: 4, OrderDate: new
Date(8377146e5),
        ShipName: 'Que Delícia', ShipCity: 'Rio de Janeiro', ShipAddress:
'Rua da Panificadora, 12',
        ShipRegion: 'RJ', ShipPostalCode: '02389-673', ShipCountry: 'Brazil',
Freight: 3.05, Verified: !1
    },
    {
        OrderID: 10262, CustomerID: 'RATTC', EmployeeID: 8, OrderDate: new
Date(8379738e5),
        ShipName: 'Rattlesnake Canyon Grocery', ShipCity: 'Albuquerque',
ShipAddress: '2817 Milton Dr.',
        ShipRegion: 'NM', ShipPostalCode: '87110', ShipCountry: 'USA',
Freight: 48.29, Verified: !0
    }
    ];

```

DATASOURCE.TSX

```

export let data: Object[] = [
    {
        OrderID: 10248, CustomerID: 'VINET', EmployeeID: 5, OrderDate: new
Date(8364186e5),
        ShipName: 'Vins et alcools Chevalier', ShipCity: 'Reims',
ShipAddress: '59 rue de l Abbaye',
        ShipRegion: 'CJ', ShipPostalCode: '51100', ShipCountry: 'France',
Freight: 32.38, Verified: !0
    },
    {
        OrderID: 10249, CustomerID: 'TOMSP', EmployeeID: 6, OrderDate: new
Date(836505e6),
        ShipName: 'Toms Spezialitäten', ShipCity: 'Münster', ShipAddress:
'Luisenstr. 48',
        ShipRegion: 'CJ', ShipPostalCode: '44087', ShipCountry: 'Germany',
Freight: 11.61, Verified: !1
    },
    {
        OrderID: 10250, CustomerID: 'HANAR', EmployeeID: 4, OrderDate: new
Date(8367642e5),
        ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
'Rua do Paço, 67',
        ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
Freight: 65.83, Verified: !0
    },
    {
        OrderID: 10251, CustomerID: 'VICTE', EmployeeID: 3, OrderDate: new
Date(8367642e5),
        ShipName: 'Victuailles en stock', ShipCity: 'Lyon', ShipAddress: '2,
rue du Commerce',

```

```

        ShipRegion: 'CJ', ShipPostalCode: '69004', ShipCountry: 'France',
        Freight: 41.34, Verified: !0
    },
    {
        OrderID: 10252, CustomerID: 'SUPRD', EmployeeID: 4, OrderDate: new
        Date(8368506e5),
        ShipName: 'Suprêmes délices', ShipCity: 'Charleroi', ShipAddress:
        'Boulevard Tirou, 255',
        ShipRegion: 'CJ', ShipPostalCode: 'B-6000', ShipCountry: 'Belgium',
        Freight: 51.3, Verified: !0
    },
    {
        OrderID: 10253, CustomerID: 'HANAR', EmployeeID: 3, OrderDate: new
        Date(836937e6),
        ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
        'Rua do Paço, 67',
        ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
        Freight: 58.17, Verified: !0
    },
    {
        OrderID: 10254, CustomerID: 'CHOPS', EmployeeID: 5, OrderDate: new
        Date(8370234e5),
        ShipName: 'Chop-suey Chinese', ShipCity: 'Bern', ShipAddress:
        'Hauptstr. 31',
        ShipRegion: 'CJ', ShipPostalCode: '3012', ShipCountry: 'Switzerland',
        Freight: 22.98, Verified: !1
    },
    {
        OrderID: 10255, CustomerID: 'RICSU', EmployeeID: 9, OrderDate: new
        Date(8371098e5),
        ShipName: 'Richter Supermarkt', ShipCity: 'Genève', ShipAddress:
        'Starenweg 5',
        ShipRegion: 'CJ', ShipPostalCode: '1204', ShipCountry: 'Switzerland',
        Freight: 148.33, Verified: !0
    },
    {
        OrderID: 10256, CustomerID: 'WELLI', EmployeeID: 3, OrderDate: new
        Date(837369e6),
        ShipName: 'Wellington Importadora', ShipCity: 'Resende', ShipAddress:
        'Rua do Mercado, 12',
        ShipRegion: 'SP', ShipPostalCode: '08737-363', ShipCountry: 'Brazil',
        Freight: 13.97, Verified: !1
    },
    {
        OrderID: 10257, CustomerID: 'HILAA', EmployeeID: 4, OrderDate: new
        Date(8374554e5),
        ShipName: 'HILARION-Abastos', ShipCity: 'San Cristóbal', ShipAddress:
        'Carrera 22 con Ave. Carlos Soublette #8-35',
        ShipRegion: 'Táchira', ShipPostalCode: '5022', ShipCountry:
        'Venezuela', Freight: 81.91, Verified: !0
    },
    {
        OrderID: 10258, CustomerID: 'ERNSH', EmployeeID: 1, OrderDate: new
        Date(8375418e5),
        ShipName: 'Ernst Handel', ShipCity: 'Graz', ShipAddress: 'Kirchgasse
        6',

```

```

        ShipRegion: 'CJ', ShipPostalCode: '8010', ShipCountry: 'Austria',
        Freight: 140.51, Verified: !0
    },
    {
        OrderID: 10259, CustomerID: 'CENTC', EmployeeID: 4, OrderDate: new
        Date(8376282e5),
        ShipName: 'Centro comercial Moctezuma', ShipCity: 'México D.F.',
        ShipAddress: 'Sierras de Granada 9993',
        ShipRegion: 'CJ', ShipPostalCode: '05022', ShipCountry: 'Mexico',
        Freight: 3.25, Verified: !1
    },
    {
        OrderID: 10260, CustomerID: 'OTTIK', EmployeeID: 4, OrderDate: new
        Date(8377146e5),
        ShipName: 'Ottilies Käseladen', ShipCity: 'Köln', ShipAddress:
        'Mehrheimerstr. 369',
        ShipRegion: 'CJ', ShipPostalCode: '50739', ShipCountry: 'Germany',
        Freight: 55.09, Verified: !0
    },
    {
        OrderID: 10261, CustomerID: 'QUEDE', EmployeeID: 4, OrderDate: new
        Date(8377146e5),
        ShipName: 'Que Delícia', ShipCity: 'Rio de Janeiro', ShipAddress:
        'Rua da Panificadora, 12',
        ShipRegion: 'RJ', ShipPostalCode: '02389-673', ShipCountry: 'Brazil',
        Freight: 3.05, Verified: !1
    },
    {
        OrderID: 10262, CustomerID: 'RATTC', EmployeeID: 8, OrderDate: new
        Date(8379738e5),
        ShipName: 'Rattlesnake Canyon Grocery', ShipCity: 'Albuquerque',
        ShipAddress: '2817 Milton Dr.',
        ShipRegion: 'NM', ShipPostalCode: '87110', ShipCountry: 'USA',
        Freight: 48.29, Verified: !0
    }
    ]};

```

Sort

The data can be ordered either in ascending or descending using [sortBy](#) method of [Query](#) class.

APP.JSX

```

import { DataManager, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { Row } from './rowTemplate';
export default class App extends React.Component {
    result = new DataManager(data).executeLocal(new Query()
        .sortBy('CustomerID').take(8));
    items = this.result.map((row, index) => (
        <Row key={index} {...row} />
    ));
    render() {
        return (
            <table id='datatable' className='e-table'>
                <thead>
                    <tr><th>Order ID</th><th>Customer ID</th><th>Employee
                    ID</th></tr>

```

```

        </thead>
        <tbody>{this.items}</tbody>
    </table>);
    }
}

```

APP.TSX

```

import { DataManager, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { IOrders } from './orders';
import { Row } from './rowTemplate';
export default class App extends React.Component<{}, {}> {
    public result: IOrders[] = new DataManager(data).executeLocal(new
    Query().sortBy('CustomerID').take(8)) as IOrders[];
    public items: React.ReactNode[] = this.result.map((row: IOrders, index)
=> (
        <Row key={index} {...row} />
    ));
    public render() {
        return (
            <table id='datatable' className='e-table'>
                <thead>
                    <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
                </thead>
                <tbody>{this.items}</tbody>
            </table>
        );
    }
}

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```

export interface IOrders {
    OrderID: number;
    EmployeeID: number;
    CustomerID: string;
    Order_Details: object[];
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
export class Row extends React.Component {
    render() {
        const item = this.props;
        return <tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>

```

```

                <td>{item.EmployeeID}</td>
            </tr>;
        }
    }
}

```

ROWTEMPLATE.TSX

```

import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
    public render() {
        const item: IOrders = this.props as IOrders;
        return <tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>
    }
}

```

APP.CSS

```

#datatable {
    border: solid 1px #e0e0e0;
    border-collapse: collapse;
    font-family: Roboto;
}
#datatable td,
#datatable th {
    border: solid #e0e0e0;
    border-width: 1px 0 0;
    display: table-cell;
    font-size: 14px;
    line-height: 20px;
    overflow: hidden;
    padding: 8px 21px;
    vertical-align: middle;
    white-space: nowrap;
    width: auto;
}

```

DATASOURCE.JSX

```

export let data = [
    {
        OrderID: 10248, CustomerID: 'VINET', EmployeeID: 5, OrderDate: new
Date(8364186e5),
        ShipName: 'Vins et alcools Chevalier', ShipCity: 'Reims',
        ShipAddress: '59 rue de l Abbaye',
        ShipRegion: 'CJ', ShipPostalCode: '51100', ShipCountry: 'France',
        Freight: 32.38, Verified: !0
    },
    {
        OrderID: 10249, CustomerID: 'TOMSP', EmployeeID: 6, OrderDate: new
Date(836505e6),
    }
]

```

```

        ShipName: 'Toms Spezialitäten', ShipCity: 'Münster', ShipAddress:
        'Luisenstr. 48',
        ShipRegion: 'CJ', ShipPostalCode: '44087', ShipCountry: 'Germany',
        Freight: 11.61, Verified: !1
    },
    {
        OrderID: 10250, CustomerID: 'HANAR', EmployeeID: 4, OrderDate: new
        Date(8367642e5),
        ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
        'Rua do Paço, 67',
        ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
        Freight: 65.83, Verified: !0
    },
    {
        OrderID: 10251, CustomerID: 'VICTE', EmployeeID: 3, OrderDate: new
        Date(8367642e5),
        ShipName: 'Victuailles en stock', ShipCity: 'Lyon', ShipAddress: '2,
        rue du Commerce',
        ShipRegion: 'CJ', ShipPostalCode: '69004', ShipCountry: 'France',
        Freight: 41.34, Verified: !0
    },
    {
        OrderID: 10252, CustomerID: 'SUPRD', EmployeeID: 4, OrderDate: new
        Date(8368506e5),
        ShipName: 'Suprêmes délices', ShipCity: 'Charleroi', ShipAddress:
        'Boulevard Tirou, 255',
        ShipRegion: 'CJ', ShipPostalCode: 'B-6000', ShipCountry: 'Belgium',
        Freight: 51.3, Verified: !0
    },
    {
        OrderID: 10253, CustomerID: 'HANAR', EmployeeID: 3, OrderDate: new
        Date(836937e6),
        ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
        'Rua do Paço, 67',
        ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
        Freight: 58.17, Verified: !0
    },
    {
        OrderID: 10254, CustomerID: 'CHOPS', EmployeeID: 5, OrderDate: new
        Date(8370234e5),
        ShipName: 'Chop-suey Chinese', ShipCity: 'Bern', ShipAddress:
        'Hauptstr. 31',
        ShipRegion: 'CJ', ShipPostalCode: '3012', ShipCountry: 'Switzerland',
        Freight: 22.98, Verified: !1
    },
    {
        OrderID: 10255, CustomerID: 'RICSU', EmployeeID: 9, OrderDate: new
        Date(8371098e5),
        ShipName: 'Richter Supermarkt', ShipCity: 'Genève', ShipAddress:
        'Starenweg 5',
        ShipRegion: 'CJ', ShipPostalCode: '1204', ShipCountry: 'Switzerland',
        Freight: 148.33, Verified: !0
    },
    {
        OrderID: 10256, CustomerID: 'WELLI', EmployeeID: 3, OrderDate: new
        Date(837369e6),

```

```

        ShipName: 'Wellington Importadora', ShipCity: 'Resende', ShipAddress:
        'Rua do Mercado, 12',
        ShipRegion: 'SP', ShipPostalCode: '08737-363', ShipCountry: 'Brazil',
        Freight: 13.97, Verified: !1
    },
    {
        OrderID: 10257, CustomerID: 'HILAA', EmployeeID: 4, OrderDate: new
        Date(8374554e5),
        ShipName: 'HILARION-Abastos', ShipCity: 'San Cristóbal', ShipAddress:
        'Carrera 22 con Ave. Carlos Soublette #8-35',
        ShipRegion: 'Táchira', ShipPostalCode: '5022', ShipCountry:
        'Venezuela', Freight: 81.91, Verified: !0
    },
    {
        OrderID: 10258, CustomerID: 'ERNSH', EmployeeID: 1, OrderDate: new
        Date(8375418e5),
        ShipName: 'Ernst Handel', ShipCity: 'Graz', ShipAddress: 'Kirchgasse
        6',
        ShipRegion: 'CJ', ShipPostalCode: '8010', ShipCountry: 'Austria',
        Freight: 140.51, Verified: !0
    },
    {
        OrderID: 10259, CustomerID: 'CENTC', EmployeeID: 4, OrderDate: new
        Date(8376282e5),
        ShipName: 'Centro comercial Moctezuma', ShipCity: 'México D.F.',
        ShipAddress: 'Sierras de Granada 9993',
        ShipRegion: 'CJ', ShipPostalCode: '05022', ShipCountry: 'Mexico',
        Freight: 3.25, Verified: !1
    },
    {
        OrderID: 10260, CustomerID: 'OTTIK', EmployeeID: 4, OrderDate: new
        Date(8377146e5),
        ShipName: 'Ottilies Käseladen', ShipCity: 'Köln', ShipAddress:
        'Mehrheimerstr. 369',
        ShipRegion: 'CJ', ShipPostalCode: '50739', ShipCountry: 'Germany',
        Freight: 55.09, Verified: !0
    },
    {
        OrderID: 10261, CustomerID: 'QUEDE', EmployeeID: 4, OrderDate: new
        Date(8377146e5),
        ShipName: 'Que Delícia', ShipCity: 'Rio de Janeiro', ShipAddress:
        'Rua da Panificadora, 12',
        ShipRegion: 'RJ', ShipPostalCode: '02389-673', ShipCountry: 'Brazil',
        Freight: 3.05, Verified: !1
    },
    {
        OrderID: 10262, CustomerID: 'RATTC', EmployeeID: 8, OrderDate: new
        Date(8379738e5),
        ShipName: 'Rattlesnake Canyon Grocery', ShipCity: 'Albuquerque',
        ShipAddress: '2817 Milton Dr.',
        ShipRegion: 'NM', ShipPostalCode: '87110', ShipCountry: 'USA',
        Freight: 48.29, Verified: !0
    }
    ]};

```

DATASOURCE.TSX

```

export let data: Object[] = [
  {
    OrderID: 10248, CustomerID: 'VINET', EmployeeID: 5, OrderDate: new
Date(8364186e5),
    ShipName: 'Vins et alcools Chevalier', ShipCity: 'Reims',
    ShipAddress: '59 rue de l Abbaye',
    ShipRegion: 'CJ', ShipPostalCode: '51100', ShipCountry: 'France',
    Freight: 32.38, Verified: !0
  },
  {
    OrderID: 10249, CustomerID: 'TOMSP', EmployeeID: 6, OrderDate: new
Date(836505e6),
    ShipName: 'Toms Spezialitäten', ShipCity: 'Münster', ShipAddress:
'Luisenstr. 48',
    ShipRegion: 'CJ', ShipPostalCode: '44087', ShipCountry: 'Germany',
    Freight: 11.61, Verified: !1
  },
  {
    OrderID: 10250, CustomerID: 'HANAR', EmployeeID: 4, OrderDate: new
Date(8367642e5),
    ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
'Rua do Paço, 67',
    ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
    Freight: 65.83, Verified: !0
  },
  {
    OrderID: 10251, CustomerID: 'VICTE', EmployeeID: 3, OrderDate: new
Date(8367642e5),
    ShipName: 'Victuailles en stock', ShipCity: 'Lyon', ShipAddress: '2,
rue du Commerce',
    ShipRegion: 'CJ', ShipPostalCode: '69004', ShipCountry: 'France',
    Freight: 41.34, Verified: !0
  },
  {
    OrderID: 10252, CustomerID: 'SUPRD', EmployeeID: 4, OrderDate: new
Date(8368506e5),
    ShipName: 'Suprêmes délices', ShipCity: 'Charleroi', ShipAddress:
'Boulevard Tirou, 255',
    ShipRegion: 'CJ', ShipPostalCode: 'B-6000', ShipCountry: 'Belgium',
    Freight: 51.3, Verified: !0
  },
  {
    OrderID: 10253, CustomerID: 'HANAR', EmployeeID: 3, OrderDate: new
Date(836937e6),
    ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
'Rua do Paço, 67',
    ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
    Freight: 58.17, Verified: !0
  },
  {
    OrderID: 10254, CustomerID: 'CHOPS', EmployeeID: 5, OrderDate: new
Date(8370234e5),
    ShipName: 'Chop-suey Chinese', ShipCity: 'Bern', ShipAddress:
'Hauptstr. 31',
    ShipRegion: 'CJ', ShipPostalCode: '3012', ShipCountry: 'Switzerland',
    Freight: 22.98, Verified: !1
  },

```



```

{
    OrderID: 10255, CustomerID: 'RICSU', EmployeeID: 9, OrderDate: new
Date(8371098e5),
    ShipName: 'Richter Supermarkt', ShipCity: 'Genève', ShipAddress:
'Starenweg 5',
    ShipRegion: 'CJ', ShipPostalCode: '1204', ShipCountry: 'Switzerland',
Freight: 148.33, Verified: !0
},
{
    OrderID: 10256, CustomerID: 'WELLI', EmployeeID: 3, OrderDate: new
Date(837369e6),
    ShipName: 'Wellington Importadora', ShipCity: 'Resende', ShipAddress:
'Rua do Mercado, 12',
    ShipRegion: 'SP', ShipPostalCode: '08737-363', ShipCountry: 'Brazil',
Freight: 13.97, Verified: !1
},
{
    OrderID: 10257, CustomerID: 'HILAA', EmployeeID: 4, OrderDate: new
Date(8374554e5),
    ShipName: 'HILARION-Abastos', ShipCity: 'San Cristóbal', ShipAddress:
'Carrera 22 con Ave. Carlos Soublette #8-35',
    ShipRegion: 'Táchira', ShipPostalCode: '5022', ShipCountry:
'Venezuela', Freight: 81.91, Verified: !0
},
{
    OrderID: 10258, CustomerID: 'ERNSH', EmployeeID: 1, OrderDate: new
Date(8375418e5),
    ShipName: 'Ernst Handel', ShipCity: 'Graz', ShipAddress: 'Kirchgasse
6',
    ShipRegion: 'CJ', ShipPostalCode: '8010', ShipCountry: 'Austria',
Freight: 140.51, Verified: !0
},
{
    OrderID: 10259, CustomerID: 'CENTC', EmployeeID: 4, OrderDate: new
Date(8376282e5),
    ShipName: 'Centro comercial Moctezuma', ShipCity: 'México D.F.',
ShipAddress: 'Sierras de Granada 9993',
    ShipRegion: 'CJ', ShipPostalCode: '05022', ShipCountry: 'Mexico',
Freight: 3.25, Verified: !1
},
{
    OrderID: 10260, CustomerID: 'OTTIK', EmployeeID: 4, OrderDate: new
Date(8377146e5),
    ShipName: 'Ottilies Käseladen', ShipCity: 'Köln', ShipAddress:
'Mehrheimerstr. 369',
    ShipRegion: 'CJ', ShipPostalCode: '50739', ShipCountry: 'Germany',
Freight: 55.09, Verified: !0
},
{
    OrderID: 10261, CustomerID: 'QUEDE', EmployeeID: 4, OrderDate: new
Date(8377146e5),
    ShipName: 'Que Delícia', ShipCity: 'Rio de Janeiro', ShipAddress:
'Rua da Panificadora, 12',
    ShipRegion: 'RJ', ShipPostalCode: '02389-673', ShipCountry: 'Brazil',
Freight: 3.05, Verified: !1
},
{

```

```

        OrderID: 10262, CustomerID: 'RATTC', EmployeeID: 8, OrderDate: new
Date(8379738e5),
        ShipName: 'Rattlesnake Canyon Grocery', ShipCity: 'Albuquerque',
ShipAddress: '2817 Milton Dr.',
        ShipRegion: 'NM', ShipPostalCode: '87110', ShipCountry: 'USA',
Freight: 48.29, Verified: !0
    }];

```

Page

The [page](#) method of the Query class is used to get range of data based on the page number and the total page size.

APP.JSX

```

import { DataManager, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { Row } from './rowTemplate';
export default class App extends React.Component {
    result = new DataManager(data).executeLocal(new Query()
        .page(1, 8));
    items = this.result.map((row, index) => (
        <Row key={index} {...row} />
    ));
    render() {
        return (<table id='datatable' className='e-table'>
            <thead>
                <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
            </thead>
            <tbody>{this.items}</tbody>
        </table>);
    }
}

```

APP.TSX

```

import { DataManager, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { IOrders } from './orders';
import { Row } from './rowTemplate';
export default class App extends React.Component<{}, {}>{
    public result: IOrders[] = new DataManager(data).executeLocal(new Query()
        .page(1, 8)) as IOrders[];
    public items: React.ReactNode[] = this.result.map((row: IOrders,
index) => (
        <Row key={index} {...row} />
    ));
    public render() {
        return (<table id='datatable' className='e-table'>
            <thead>
                <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
            </thead>

```

```
        <tbody>{this.items}</tbody>
    </table>);
    }
}
```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```
export interface IOrders {
    OrderID: number;
    EmployeeID: number;
    CustomerID: string;
    Order_Details: object[];
}
```

ROWTEMPLATE.JSX

```
import * as React from 'react';
export class Row extends React.Component {
    render() {
        const item = this.props;
        return <tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>;
    }
}
```

ROWTEMPLATE.TSX

```
import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
    public render() {
        const item: IOrders = this.props as IOrders;
        return <tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>
    }
}
```

APP.CSS

```
#datatable {
    border: solid 1px #e0e0e0;
    border-collapse: collapse;
    font-family: Roboto;
}
```

```

#datatable td,
#datatable th {
    border: solid #e0e0e0;
    border-width: 1px 0 0;
    display: table-cell;
    font-size: 14px;
    line-height: 20px;
    overflow: hidden;
    padding: 8px 21px;
    vertical-align: middle;
    white-space: nowrap;
    width: auto;
}

```

DATASOURCE.JSX

```

export let data = [
  {
    OrderID: 10248, CustomerID: 'VINET', EmployeeID: 5, OrderDate: new
Date(8364186e5),
    ShipName: 'Vins et alcools Chevalier', ShipCity: 'Reims',
    ShipAddress: '59 rue de l Abbaye',
    ShipRegion: 'CJ', ShipPostalCode: '51100', ShipCountry: 'France',
    Freight: 32.38, Verified: !0
  },
  {
    OrderID: 10249, CustomerID: 'TOMSP', EmployeeID: 6, OrderDate: new
Date(836505e6),
    ShipName: 'Toms Spezialitäten', ShipCity: 'Münster', ShipAddress:
'Luisenstr. 48',
    ShipRegion: 'CJ', ShipPostalCode: '44087', ShipCountry: 'Germany',
    Freight: 11.61, Verified: !1
  },
  {
    OrderID: 10250, CustomerID: 'HANAR', EmployeeID: 4, OrderDate: new
Date(8367642e5),
    ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
'Rua do Paço, 67',
    ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
    Freight: 65.83, Verified: !0
  },
  {
    OrderID: 10251, CustomerID: 'VICTE', EmployeeID: 3, OrderDate: new
Date(8367642e5),
    ShipName: 'Victuailles en stock', ShipCity: 'Lyon', ShipAddress: '2,
rue du Commerce',
    ShipRegion: 'CJ', ShipPostalCode: '69004', ShipCountry: 'France',
    Freight: 41.34, Verified: !0
  },
  {
    OrderID: 10252, CustomerID: 'SUPRD', EmployeeID: 4, OrderDate: new
Date(8368506e5),
    ShipName: 'Suprêmes délices', ShipCity: 'Charleroi', ShipAddress:
'Boulevard Tirou, 255',
    ShipRegion: 'CJ', ShipPostalCode: 'B-6000', ShipCountry: 'Belgium',
    Freight: 51.3, Verified: !0
  }
]

```

```

    },
    {
        OrderID: 10253, CustomerID: 'HANAR', EmployeeID: 3, OrderDate: new
Date(836937e6),
        ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
'Rua do Paço, 67',
        ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
Freight: 58.17, Verified: !0
    },
    {
        OrderID: 10254, CustomerID: 'CHOPS', EmployeeID: 5, OrderDate: new
Date(8370234e5),
        ShipName: 'Chop-suey Chinese', ShipCity: 'Bern', ShipAddress:
'Hauptstr. 31',
        ShipRegion: 'CJ', ShipPostalCode: '3012', ShipCountry: 'Switzerland',
Freight: 22.98, Verified: !1
    },
    {
        OrderID: 10255, CustomerID: 'RICSU', EmployeeID: 9, OrderDate: new
Date(8371098e5),
        ShipName: 'Richter Supermarkt', ShipCity: 'Genève', ShipAddress:
'Starenweg 5',
        ShipRegion: 'CJ', ShipPostalCode: '1204', ShipCountry: 'Switzerland',
Freight: 148.33, Verified: !0
    },
    {
        OrderID: 10256, CustomerID: 'WELLI', EmployeeID: 3, OrderDate: new
Date(837369e6),
        ShipName: 'Wellington Importadora', ShipCity: 'Resende', ShipAddress:
'Rua do Mercado, 12',
        ShipRegion: 'SP', ShipPostalCode: '08737-363', ShipCountry: 'Brazil',
Freight: 13.97, Verified: !1
    },
    {
        OrderID: 10257, CustomerID: 'HILAA', EmployeeID: 4, OrderDate: new
Date(8374554e5),
        ShipName: 'HILARION-Abastos', ShipCity: 'San Cristóbal', ShipAddress:
'Carrera 22 con Ave. Carlos Soublette #8-35',
        ShipRegion: 'Táchira', ShipPostalCode: '5022', ShipCountry:
'Venezuela', Freight: 81.91, Verified: !0
    },
    {
        OrderID: 10258, CustomerID: 'ERNSH', EmployeeID: 1, OrderDate: new
Date(8375418e5),
        ShipName: 'Ernst Handel', ShipCity: 'Graz', ShipAddress: 'Kirchgasse
6',
        ShipRegion: 'CJ', ShipPostalCode: '8010', ShipCountry: 'Austria',
Freight: 140.51, Verified: !0
    },
    {
        OrderID: 10259, CustomerID: 'CENTC', EmployeeID: 4, OrderDate: new
Date(8376282e5),
        ShipName: 'Centro comercial Moctezuma', ShipCity: 'México D.F.',
ShipAddress: 'Sierras de Granada 9993',
        ShipRegion: 'CJ', ShipPostalCode: '05022', ShipCountry: 'Mexico',
Freight: 3.25, Verified: !1
    },
    },

```

```

{
    OrderID: 10260, CustomerID: 'OTTIK', EmployeeID: 4, OrderDate: new
Date(8377146e5),
    ShipName: 'Ottilies Käseladen', ShipCity: 'Köln', ShipAddress:
'Mehrheimerstr. 369',
    ShipRegion: 'CJ', ShipPostalCode: '50739', ShipCountry: 'Germany',
Freight: 55.09, Verified: !0
},
{
    OrderID: 10261, CustomerID: 'QUEDE', EmployeeID: 4, OrderDate: new
Date(8377146e5),
    ShipName: 'Que Delícia', ShipCity: 'Rio de Janeiro', ShipAddress:
'Rua da Panificadora, 12',
    ShipRegion: 'RJ', ShipPostalCode: '02389-673', ShipCountry: 'Brazil',
Freight: 3.05, Verified: !1
},
{
    OrderID: 10262, CustomerID: 'RATTC', EmployeeID: 8, OrderDate: new
Date(8379738e5),
    ShipName: 'Rattlesnake Canyon Grocery', ShipCity: 'Albuquerque',
ShipAddress: '2817 Milton Dr.',
    ShipRegion: 'NM', ShipPostalCode: '87110', ShipCountry: 'USA',
Freight: 48.29, Verified: !0
}
];

```

DATASOURCE.TSX

```

export let data: Object[] = [
{
    OrderID: 10248, CustomerID: 'VINET', EmployeeID: 5, OrderDate: new
Date(8364186e5),
    ShipName: 'Vins et alcools Chevalier', ShipCity: 'Reims',
ShipAddress: '59 rue de l Abbaye',
    ShipRegion: 'CJ', ShipPostalCode: '51100', ShipCountry: 'France',
Freight: 32.38, Verified: !0
},
{
    OrderID: 10249, CustomerID: 'TOMSP', EmployeeID: 6, OrderDate: new
Date(836505e6),
    ShipName: 'Toms Spezialitäten', ShipCity: 'Münster', ShipAddress:
'Luisenstr. 48',
    ShipRegion: 'CJ', ShipPostalCode: '44087', ShipCountry: 'Germany',
Freight: 11.61, Verified: !1
},
{
    OrderID: 10250, CustomerID: 'HANAR', EmployeeID: 4, OrderDate: new
Date(8367642e5),
    ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
'Rua do Paço, 67',
    ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
Freight: 65.83, Verified: !0
},
{
    OrderID: 10251, CustomerID: 'VICTE', EmployeeID: 3, OrderDate: new
Date(8367642e5),

```

```

        ShipName: 'Victuailles en stock', ShipCity: 'Lyon', ShipAddress: '2,
rue du Commerce',
        ShipRegion: 'CJ', ShipPostalCode: '69004', ShipCountry: 'France',
        Freight: 41.34, Verified: !0
    },
    {
        OrderID: 10252, CustomerID: 'SUPRD', EmployeeID: 4, OrderDate: new
Date(8368506e5),
        ShipName: 'Suprêmes délices', ShipCity: 'Charleroi', ShipAddress:
'Boulevard Tirou, 255',
        ShipRegion: 'CJ', ShipPostalCode: 'B-6000', ShipCountry: 'Belgium',
        Freight: 51.3, Verified: !0
    },
    {
        OrderID: 10253, CustomerID: 'HANAR', EmployeeID: 3, OrderDate: new
Date(836937e6),
        ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
'Rua do Paço, 67',
        ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
        Freight: 58.17, Verified: !0
    },
    {
        OrderID: 10254, CustomerID: 'CHOPS', EmployeeID: 5, OrderDate: new
Date(8370234e5),
        ShipName: 'Chop-suey Chinese', ShipCity: 'Bern', ShipAddress:
'Hauptstr. 31',
        ShipRegion: 'CJ', ShipPostalCode: '3012', ShipCountry: 'Switzerland',
        Freight: 22.98, Verified: !1
    },
    {
        OrderID: 10255, CustomerID: 'RICSU', EmployeeID: 9, OrderDate: new
Date(8371098e5),
        ShipName: 'Richter Supermarkt', ShipCity: 'Genève', ShipAddress:
'Starenweg 5',
        ShipRegion: 'CJ', ShipPostalCode: '1204', ShipCountry: 'Switzerland',
        Freight: 148.33, Verified: !0
    },
    {
        OrderID: 10256, CustomerID: 'WELLI', EmployeeID: 3, OrderDate: new
Date(837369e6),
        ShipName: 'Wellington Importadora', ShipCity: 'Resende', ShipAddress:
'Rua do Mercado, 12',
        ShipRegion: 'SP', ShipPostalCode: '08737-363', ShipCountry: 'Brazil',
        Freight: 13.97, Verified: !1
    },
    {
        OrderID: 10257, CustomerID: 'HILAA', EmployeeID: 4, OrderDate: new
Date(8374554e5),
        ShipName: 'HILARION-Abastos', ShipCity: 'San Cristóbal', ShipAddress:
'Carrera 22 con Ave. Carlos Soublette #8-35',
        ShipRegion: 'Táchira', ShipPostalCode: '5022', ShipCountry:
'Venezuela', Freight: 81.91, Verified: !0
    },
    {
        OrderID: 10258, CustomerID: 'ERNSH', EmployeeID: 1, OrderDate: new
Date(8375418e5),

```

```

        ShipName: 'Ernst Handel', ShipCity: 'Graz', ShipAddress: 'Kirchgasse
6',
        ShipRegion: 'CJ', ShipPostalCode: '8010', ShipCountry: 'Austria',
        Freight: 140.51, Verified: !0
    },
    {
        OrderID: 10259, CustomerID: 'CENTC', EmployeeID: 4, OrderDate: new
Date(8376282e5),
        ShipName: 'Centro comercial Moctezuma', ShipCity: 'México D.F.',
        ShipAddress: 'Sierras de Granada 9993',
        ShipRegion: 'CJ', ShipPostalCode: '05022', ShipCountry: 'Mexico',
        Freight: 3.25, Verified: !1
    },
    {
        OrderID: 10260, CustomerID: 'OTTIK', EmployeeID: 4, OrderDate: new
Date(8377146e5),
        ShipName: 'Ottilies Käseladen', ShipCity: 'Köln', ShipAddress:
'Mehrheimerstr. 369',
        ShipRegion: 'CJ', ShipPostalCode: '50739', ShipCountry: 'Germany',
        Freight: 55.09, Verified: !0
    },
    {
        OrderID: 10261, CustomerID: 'QUEDE', EmployeeID: 4, OrderDate: new
Date(8377146e5),
        ShipName: 'Que Delícia', ShipCity: 'Rio de Janeiro', ShipAddress:
'Rua da Panificadora, 12',
        ShipRegion: 'RJ', ShipPostalCode: '02389-673', ShipCountry: 'Brazil',
        Freight: 3.05, Verified: !1
    },
    {
        OrderID: 10262, CustomerID: 'RATTC', EmployeeID: 8, OrderDate: new
Date(8379738e5),
        ShipName: 'Rattlesnake Canyon Grocery', ShipCity: 'Albuquerque',
        ShipAddress: '2817 Milton Dr.',
        ShipRegion: 'NM', ShipPostalCode: '87110', ShipCountry: 'USA',
        Freight: 48.29, Verified: !0
    }
    ]];

```

Component binding

DataManager component can be used with Syncfusion components which supports data binding.

In the following samples, the grid component is bound. To render the grid with the necessary configurations, please refer to the [React Grid getting started](#) documentation.

Local data binding

A data source can be created in-line with other Syncfusion component configuration settings.

APP.JSX

```

import { DataManager } from '@syncfusion/ej2-data';
import { ColumnDirective, ColumnsDirective, GridComponent } from
 '@syncfusion/ej2-react-grids';
import * as React from "react";
import { data } from './datasource';
export default class App extends React.Component {
    data = new DataManager(data);

```



```

render() {
    return <GridComponent dataSource={this.data}>
        <ColumnsDirective>
            <ColumnDirective field='OrderID' width='100'
textAlign="Right"/>
            <ColumnDirective field='CustomerID' width='100' />
            <ColumnDirective field='EmployeeID' width='100'
textAlign="Right"/>
            <ColumnDirective field='Freight' width='100' format="C2"
textAlign="Right"/>
            <ColumnDirective field='ShipCountry' width='100' />
        </ColumnsDirective>
    </GridComponent>;
}
}
;

```

APP.TSX

```

import { DataManager } from '@syncfusion/ej2-data';
import { ColumnDirective, ColumnsDirective, GridComponent } from
'@syncfusion/ej2-react-grids';
import * as React from "react";
import { data } from './datasource';
export default class App extends React.Component<{}, {}>{
    public data: DataManager = new DataManager(data);
    public render() {
        return <GridComponent dataSource={this.data}>
            <ColumnsDirective>
                <ColumnDirective field='OrderID' width='100'
textAlign="Right"/>
                <ColumnDirective field='CustomerID' width='100' />
                <ColumnDirective field='EmployeeID' width='100'
textAlign="Right"/>
                <ColumnDirective field='Freight' width='100' format="C2"
textAlign="Right"/>
                <ColumnDirective field='ShipCountry' width='100' />
            </ColumnsDirective>
        </GridComponent>
    }
};

```

DATASOURCE.JSX

```

export let data= [
    {
        OrderID: 10248, CustomerID: 'VINET', EmployeeID: 5, OrderDate: new
Date(8364186e5),
        ShipName: 'Vins et alcools Chevalier', ShipCity: 'Reims',
        ShipAddress: '59 rue de l Abbaye',
        ShipRegion: 'CJ', ShipPostalCode: '51100', ShipCountry: 'France',
        Freight: 32.38, Verified: !0
    },
    {
        OrderID: 10249, CustomerID: 'TOMSP', EmployeeID: 6, OrderDate: new
Date(836505e6),
    }
];

```

```

        ShipName: 'Toms Spezialitäten', ShipCity: 'Münster', ShipAddress:
        'Luisenstr. 48',
        ShipRegion: 'CJ', ShipPostalCode: '44087', ShipCountry: 'Germany',
        Freight: 11.61, Verified: !1
    },
    {
        OrderID: 10250, CustomerID: 'HANAR', EmployeeID: 4, OrderDate: new
        Date(8367642e5),
        ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
        'Rua do Paço, 67',
        ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
        Freight: 65.83, Verified: !0
    },
    {
        OrderID: 10251, CustomerID: 'VICTE', EmployeeID: 3, OrderDate: new
        Date(8367642e5),
        ShipName: 'Victuailles en stock', ShipCity: 'Lyon', ShipAddress: '2,
        rue du Commerce',
        ShipRegion: 'CJ', ShipPostalCode: '69004', ShipCountry: 'France',
        Freight: 41.34, Verified: !0
    },
    {
        OrderID: 10252, CustomerID: 'SUPRD', EmployeeID: 4, OrderDate: new
        Date(8368506e5),
        ShipName: 'Suprêmes délices', ShipCity: 'Charleroi', ShipAddress:
        'Boulevard Tirou, 255',
        ShipRegion: 'CJ', ShipPostalCode: 'B-6000', ShipCountry: 'Belgium',
        Freight: 51.3, Verified: !0
    },
    {
        OrderID: 10253, CustomerID: 'HANAR', EmployeeID: 3, OrderDate: new
        Date(836937e6),
        ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
        'Rua do Paço, 67',
        ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
        Freight: 58.17, Verified: !0
    },
    {
        OrderID: 10254, CustomerID: 'CHOPS', EmployeeID: 5, OrderDate: new
        Date(8370234e5),
        ShipName: 'Chop-suey Chinese', ShipCity: 'Bern', ShipAddress:
        'Hauptstr. 31',
        ShipRegion: 'CJ', ShipPostalCode: '3012', ShipCountry: 'Switzerland',
        Freight: 22.98, Verified: !1
    },
    {
        OrderID: 10255, CustomerID: 'RICSU', EmployeeID: 9, OrderDate: new
        Date(8371098e5),
        ShipName: 'Richter Supermarkt', ShipCity: 'Genève', ShipAddress:
        'Starenweg 5',
        ShipRegion: 'CJ', ShipPostalCode: '1204', ShipCountry: 'Switzerland',
        Freight: 148.33, Verified: !0
    },
    {
        OrderID: 10256, CustomerID: 'WELLI', EmployeeID: 3, OrderDate: new
        Date(837369e6),

```

```

        ShipName: 'Wellington Importadora', ShipCity: 'Resende', ShipAddress:
        'Rua do Mercado, 12',
        ShipRegion: 'SP', ShipPostalCode: '08737-363', ShipCountry: 'Brazil',
        Freight: 13.97, Verified: !1
    },
    {
        OrderID: 10257, CustomerID: 'HILAA', EmployeeID: 4, OrderDate: new
        Date(8374554e5),
        ShipName: 'HILARION-Abastos', ShipCity: 'San Cristóbal', ShipAddress:
        'Carrera 22 con Ave. Carlos Soublette #8-35',
        ShipRegion: 'Táchira', ShipPostalCode: '5022', ShipCountry:
        'Venezuela', Freight: 81.91, Verified: !0
    },
    {
        OrderID: 10258, CustomerID: 'ERNSH', EmployeeID: 1, OrderDate: new
        Date(8375418e5),
        ShipName: 'Ernst Handel', ShipCity: 'Graz', ShipAddress: 'Kirchgasse
        6',
        ShipRegion: 'CJ', ShipPostalCode: '8010', ShipCountry: 'Austria',
        Freight: 140.51, Verified: !0
    },
    {
        OrderID: 10259, CustomerID: 'CENTC', EmployeeID: 4, OrderDate: new
        Date(8376282e5),
        ShipName: 'Centro comercial Moctezuma', ShipCity: 'México D.F.',
        ShipAddress: 'Sierras de Granada 9993',
        ShipRegion: 'CJ', ShipPostalCode: '05022', ShipCountry: 'Mexico',
        Freight: 3.25, Verified: !1
    },
    {
        OrderID: 10260, CustomerID: 'OTTIK', EmployeeID: 4, OrderDate: new
        Date(8377146e5),
        ShipName: 'Ottilies Käseladen', ShipCity: 'Köln', ShipAddress:
        'Mehrheimerstr. 369',
        ShipRegion: 'CJ', ShipPostalCode: '50739', ShipCountry: 'Germany',
        Freight: 55.09, Verified: !0
    },
    {
        OrderID: 10261, CustomerID: 'QUEDE', EmployeeID: 4, OrderDate: new
        Date(8377146e5),
        ShipName: 'Que Delícia', ShipCity: 'Rio de Janeiro', ShipAddress:
        'Rua da Panificadora, 12',
        ShipRegion: 'RJ', ShipPostalCode: '02389-673', ShipCountry: 'Brazil',
        Freight: 3.05, Verified: !1
    },
    {
        OrderID: 10262, CustomerID: 'RATTC', EmployeeID: 8, OrderDate: new
        Date(8379738e5),
        ShipName: 'Rattlesnake Canyon Grocery', ShipCity: 'Albuquerque',
        ShipAddress: '2817 Milton Dr.',
        ShipRegion: 'NM', ShipPostalCode: '87110', ShipCountry: 'USA',
        Freight: 48.29, Verified: !0
    }
    ]};

```

DATASOURCE.TSX

```

export let data: Object[] = [
  {
    OrderID: 10248, CustomerID: 'VINET', EmployeeID: 5, OrderDate: new
    Date(8364186e5),
    ShipName: 'Vins et alcools Chevalier', ShipCity: 'Reims',
    ShipAddress: '59 rue de l Abbaye',
    ShipRegion: 'CJ', ShipPostalCode: '51100', ShipCountry: 'France',
    Freight: 32.38, Verified: !0
  },
  {
    OrderID: 10249, CustomerID: 'TOMSP', EmployeeID: 6, OrderDate: new
    Date(836505e6),
    ShipName: 'Toms Spezialitäten', ShipCity: 'Münster', ShipAddress:
    'Luisenstr. 48',
    ShipRegion: 'CJ', ShipPostalCode: '44087', ShipCountry: 'Germany',
    Freight: 11.61, Verified: !1
  },
  {
    OrderID: 10250, CustomerID: 'HANAR', EmployeeID: 4, OrderDate: new
    Date(8367642e5),
    ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
    'Rua do Paço, 67',
    ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
    Freight: 65.83, Verified: !0
  },
  {
    OrderID: 10251, CustomerID: 'VICTE', EmployeeID: 3, OrderDate: new
    Date(8367642e5),
    ShipName: 'Victuailles en stock', ShipCity: 'Lyon', ShipAddress: '2,
    rue du Commerce',
    ShipRegion: 'CJ', ShipPostalCode: '69004', ShipCountry: 'France',
    Freight: 41.34, Verified: !0
  },
  {
    OrderID: 10252, CustomerID: 'SUPRD', EmployeeID: 4, OrderDate: new
    Date(8368506e5),
    ShipName: 'Suprêmes délices', ShipCity: 'Charleroi', ShipAddress:
    'Boulevard Tirou, 255',
    ShipRegion: 'CJ', ShipPostalCode: 'B-6000', ShipCountry: 'Belgium',
    Freight: 51.3, Verified: !0
  },
  {
    OrderID: 10253, CustomerID: 'HANAR', EmployeeID: 3, OrderDate: new
    Date(836937e6),
    ShipName: 'Hanari Carnes', ShipCity: 'Rio de Janeiro', ShipAddress:
    'Rua do Paço, 67',
    ShipRegion: 'RJ', ShipPostalCode: '05454-876', ShipCountry: 'Brazil',
    Freight: 58.17, Verified: !0
  },
  {
    OrderID: 10254, CustomerID: 'CHOPS', EmployeeID: 5, OrderDate: new
    Date(8370234e5),
    ShipName: 'Chop-suey Chinese', ShipCity: 'Bern', ShipAddress:
    'Hauptstr. 31',
    ShipRegion: 'CJ', ShipPostalCode: '3012', ShipCountry: 'Switzerland',
    Freight: 22.98, Verified: !1
  },
]

```

```

{
    OrderID: 10255, CustomerID: 'RICSU', EmployeeID: 9, OrderDate: new
Date(8371098e5),
    ShipName: 'Richter Supermarkt', ShipCity: 'Genève', ShipAddress:
'Starenweg 5',
    ShipRegion: 'CJ', ShipPostalCode: '1204', ShipCountry: 'Switzerland',
Freight: 148.33, Verified: !0
},
{
    OrderID: 10256, CustomerID: 'WELLI', EmployeeID: 3, OrderDate: new
Date(837369e6),
    ShipName: 'Wellington Importadora', ShipCity: 'Resende', ShipAddress:
'Rua do Mercado, 12',
    ShipRegion: 'SP', ShipPostalCode: '08737-363', ShipCountry: 'Brazil',
Freight: 13.97, Verified: !1
},
{
    OrderID: 10257, CustomerID: 'HILAA', EmployeeID: 4, OrderDate: new
Date(8374554e5),
    ShipName: 'HILARION-Abastos', ShipCity: 'San Cristóbal', ShipAddress:
'Carrera 22 con Ave. Carlos Soublette #8-35',
    ShipRegion: 'Táchira', ShipPostalCode: '5022', ShipCountry:
'Venezuela', Freight: 81.91, Verified: !0
},
{
    OrderID: 10258, CustomerID: 'ERNSH', EmployeeID: 1, OrderDate: new
Date(8375418e5),
    ShipName: 'Ernst Handel', ShipCity: 'Graz', ShipAddress: 'Kirchgasse
6',
    ShipRegion: 'CJ', ShipPostalCode: '8010', ShipCountry: 'Austria',
Freight: 140.51, Verified: !0
},
{
    OrderID: 10259, CustomerID: 'CENTC', EmployeeID: 4, OrderDate: new
Date(8376282e5),
    ShipName: 'Centro comercial Moctezuma', ShipCity: 'México D.F.',
ShipAddress: 'Sierras de Granada 9993',
    ShipRegion: 'CJ', ShipPostalCode: '05022', ShipCountry: 'Mexico',
Freight: 3.25, Verified: !1
},
{
    OrderID: 10260, CustomerID: 'OTTIK', EmployeeID: 4, OrderDate: new
Date(8377146e5),
    ShipName: 'Ottilies Käseladen', ShipCity: 'Köln', ShipAddress:
'Mehrheimerstr. 369',
    ShipRegion: 'CJ', ShipPostalCode: '50739', ShipCountry: 'Germany',
Freight: 55.09, Verified: !0
},
{
    OrderID: 10261, CustomerID: 'QUEDE', EmployeeID: 4, OrderDate: new
Date(8377146e5),
    ShipName: 'Que Delícia', ShipCity: 'Rio de Janeiro', ShipAddress:
'Rua da Panificadora, 12',
    ShipRegion: 'RJ', ShipPostalCode: '02389-673', ShipCountry: 'Brazil',
Freight: 3.05, Verified: !1
},
{

```

```

        OrderID: 10262, CustomerID: 'RATTC', EmployeeID: 8, OrderDate: new
Date(8379738e5),
        ShipName: 'Rattlesnake Canyon Grocery', ShipCity: 'Albuquerque',
ShipAddress: '2817 Milton Dr.',
        ShipRegion: 'NM', ShipPostalCode: '87110', ShipCountry: 'USA',
Freight: 48.29, Verified: !0
    }];

```

Remote data binding

To bind remote data to Syncfusion component, you can assign a service data as an instance of [DataManager](#) to the **dataSource** property.

APP.JSX

```

import { DataManager, ODataV4Adaptor } from '@syncfusion/ej2-data';
import { ColumnDirective, ColumnsDirective, GridComponent } from
 '@syncfusion/ej2-react-grids';
import * as React from "react";
const SERVICE_URI =
 'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component {
    data = new DataManager({ url: SERVICE_URI , adaptor: new ODataV4Adaptor()
});
    render() {
        return <GridComponent dataSource={this.data}>
            <ColumnsDirective>
                <ColumnDirective field='OrderID' width='100'
textAlign="Right"/>
                <ColumnDirective field='CustomerID' width='100' />
                <ColumnDirective field='EmployeeID' width='100'
textAlign="Right"/>
                <ColumnDirective field='Freight' width='100' format="C2"
textAlign="Right"/>
                <ColumnDirective field='ShipCountry' width='100' />
            </ColumnsDirective>
        </GridComponent>;
    }
}
;

```

APP.TSX

```

import { DataManager, ODataV4Adaptor } from '@syncfusion/ej2-data';
import { ColumnDirective, ColumnsDirective, GridComponent, } from
 '@syncfusion/ej2-react-grids';
import * as React from "react";
const SERVICE_URI:string =
 'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component<{}, {}>{
    public data: DataManager = new DataManager({ url: SERVICE_URI , adaptor:
new ODataV4Adaptor() });
    public render() {
        return <GridComponent dataSource={this.data}>
            <ColumnsDirective>

```

```

        <ColumnDirective field='OrderID' width='100'
textAlign="Right"/>
        <ColumnDirective field='CustomerID' width='100' />
        <ColumnDirective field='EmployeeID' width='100'
textAlign="Right"/>
        <ColumnDirective field='Freight' width='100' format="C2"
textAlign="Right"/>
        <ColumnDirective field='ShipCountry' width='100' />
    </ColumnsDirective>
</GridComponent>
    }
};

```

Data binding in React Data component

[DataManager](#) supports both RESTful JSON data services binding and local JavaScript object array binding.

Local data binding

[DataManager](#) can be bound to local data source by assigning the array of JavaScript objects to the `json` property or simply passing them to the constructor while instantiating. Now the JavaScript object array can be queried and manipulated.

APP.JSX

```

import { DataManager, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { Row } from './rowTemplate';
export default class App extends React.Component {
    result = new DataManager(data).executeLocal(new Query().take(8));
    items = this.result.map((row, index) => (
        <Row key={index} {...row} />
    ));
    render() {
        return (
            <table id='datatable' className='e-table'>
                <thead>
                    <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
                </thead>
                <tbody>{this.items}</tbody>
            </table>);
    }
}

```

APP.TSX

```

import { DataManager, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { Row } from './rowTemplate';
export default class App extends React.Component<{}, {}> {

    public result: Object[] = new DataManager(data).executeLocal(new
Query().take(8));
}

```

```

    public items: React.ReactNode[] = this.result.map((row: object, index:
number) => (
        <Row key={index} {...row} />
    ));
    public render() {
        return (
            <table id='datatable' className='e-table'>
                <thead>
                    <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
                </thead>
                <tbody>{this.items}</tbody>
            </table>
        );
    }
}

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```

export interface IOrders {
    OrderID: number;
    EmployeeID: number;
    CustomerID: string;
    Order_Details: object[];
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
export class Row extends React.Component {
    render() {
        const item = this.props;
        return <tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>;
    }
}

```

ROWTEMPLATE.TSX

```

import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
    public render() {
        const item: IOrders = this.props as IOrders;
        return <tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>

```



```

        </tr>
    }
}

```

Remote data binding

[DataManager](#) can be bound to remote data source by assigning service end point URL to the `url` property. With the provided `url`, the [DataManager](#) handles all communication with the data server with help of queries.

When querying data, the [DataManager](#) will convert the query object, [Query](#) into server request after calling `executeQuery` and waits for the server response(JSON format).

APP.JSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, Query, ODataV4Adaptor } from '@syncfusion/ej2-data';
import * as React from 'react';
import { Row } from './rowTemplate';
const SERVICE_URI =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component {
    constructor(props) {
        super(props);
        this.state = { items: [] };
    }
    componentDidMount() {
        new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor()
    }).executeQuery(new Query().take(8))
        .then((e) => {
            const res = e.result.map((row) => <Row key={row.OrderID}
{...row} />);
            this.setState({
                items: res,
            });
        });
    }
    render() {
        return (
            <table id='datatable' className='e-table'>
                <thead>
                    <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
                </thead>
                <tbody>{getValue('items', this.state)}</tbody>
            </table>);
    }
}

```

APP.TSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, Query, ReturnOption, ODataV4Adaptor } from
 '@syncfusion/ej2-data';
import * as React from 'react';
import { IOOrders } from './orders';
import { Row } from './rowTemplate';

```

```

const SERVICE_URI: string =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component<{}, {}>{
  constructor(props: object) {
    super(props);
    this.state = { items: [] };
  }
  componentDidMount() {
    new DataManager({ url: SERVICE_URI ,adaptor: new ODataV4Adaptor()
  }).executeQuery(new Query().take(8))
    .then((e: ReturnOption) => {
      const res = (e.result as IOrders[]).map((row: IOrders, index:
number) => (
        <Row key={row.OrderID} {...row} />
      ));
      this.setState({
        items: res
      });
    });
  }
  public render() {
    return <table id='datatable' className='e-table'>
      <thead>
        <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
      </thead>
      <tbody>{ getValue('items', this.state) }</tbody>
    </table>
  }
}

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```

export interface IOrders {
  OrderID: number;
  EmployeeID: number;
  CustomerID: string;
  Order_Details: object[];
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
export class Row extends React.Component {
  render() {
    const item = this.props;
    return <tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>;
  }
}

```

```
}
}
```

ROWTEMPLATE.TSX

```
import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
  public render() {
    const item: IOrders = this.props as IOrders;
    return (<tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>)
  }
}
```

The queried data will not be cached locally unless offline mode is enabled.

See Also

- [Binding with OData service](#)
- [Binding with ODataV4 service](#)
- [Binding with Web API](#)
- [How to write custom adaptor](#)
- [How to work in offline mode](#)
- [How to send additional parameters](#)
- [How to add custom request headers](#)

Adaptors in React Data component

Each data source or remote service uses different way in accepting request and sending back the response. **DataManager** cannot anticipate every way a data source

works. To tackle this problem the **DataManager** uses the adaptor concept to communicate with particular data source.

For local data sources, the role of the data adaptor is to query the JavaScript object array based on the **Query** object and manipulate them.

When comes with remote datasource, the data adaptor is used to send the request that the server can understand and process the server response.

The adaptor can be assigned using the **adaptor** property of the **DataManager**.

Json adaptor

JsonAdaptor is used to query and manipulate JavaScript object array.

APP.JSX

```
import { DataManager, JsonAdaptor, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
```

```
import { data } from './datasource';
import { Row } from './rowTemplate';
export default class App extends React.Component {
  result = new DataManager({ json: data, adaptor: new JsonAdaptor })
    .executeLocal(new Query().take(8));
  items = this.result.map((row, index) => (
    <Row key={index} {...row} />
  ));
  render() {
    return (
      <table id='datatable' className='e-table'>
        <thead>
          <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
        </thead>
        <tbody>{this.items}</tbody>
      </table>);
  }
}
```

APP.TSX

```
import { DataManager, JsonAdaptor, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { Row } from './rowTemplate';
export default class App extends React.Component<{}, {}> {
  public result: Object[] = new DataManager({ json: data, adaptor: new
JsonAdaptor() })
    .executeLocal(new Query().take(8));

  public items: React.ReactElement[] = this.result.map((row: object, index)
=> (
    <Row key={index} {...row} />
  ));
  public render() {
    return (
      <table id='datatable' className='e-table'>
        <thead>
          <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
        </thead>
        <tbody>{this.items}</tbody>
      </table>
    );
  }
}
```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```
export interface IOrders {
  OrderID: number;
```

```

EmployeeID: number;
CustomerID: string;
Order_Details: object[];
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
export class Row extends React.Component {
  render() {
    const item = this.props;
    return (<tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>);
  }
}

```

ROWTEMPLATE.TSX

```

import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
  public render() {
    const item: IOrders = this.props as IOrders;
    return (<tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>);
  }
}

```

[Url adaptor](#)

UrlAdaptor act as the base adaptor for interacting with remote data services. Most of the built-in adaptors are derived from the **UrlAdaptor**.

`ts

```

import { DataManager, Query, ReturnOption, UrlAdaptor } from '@syncfusion/ej2-data';
const SERVICE_URI: string = 'https://ej2services.syncfusion.com/react/development/api/UrlDataSource';
new DataManager({
  adaptor: new UrlAdaptor,
  url: SERVICE_URI
}).executeQuery(new Query().take(8)).then((e: ReturnOption) => {
  // e.result will contain the records
});
`

```

UrlAdaptor expects response as a JSON object with properties **result** and **count** which contains the collection of entities and the total number of records respectively.

The sample response object should be as follows,

```
,
{
  "result": [{..}, {..}, {..}, ...],
  "count": 67
},
```

OData adaptor

OData is standardized protocol for creating and consuming data. You can retrieve data from OData service using **DataManager**. The **ODataAdaptor** helps you to interact with OData service. You can refer to the following code example of remote Data binding using OData service.

```
`ts
import { DataManager, Query, ReturnOption, ODataAdaptor } from '@syncfusion/ej2-data';
const SERVICE_URI: string = 'https://services.odata.org/V3/Northwind/Northwind.svc/Orders/';
new DataManager({
  adaptor: new ODataAdaptor,
  url: SERVICE_URI
}).executeQuery(new Query().take(8)).then((e: ReturnOption) => {
  // e.result will contain the records
});
,
```

ODataAdaptor expects JSON response from the server and the response object should contain properties **Items**, **Result** and **Count** whose values are collection of entities and total count of the entities respectively.

The sample response object should look like below.

```
,
{
  Result: [{..}, {..}, {..}, ...],
  Items: [{..}, {..}, {..}, ...],
  Count: 830
},
```

By default, **ODataAdaptor** is used by **DataManager**.

ODataV4 adaptor

The ODataV4 is an improved version of OData protocols and the **DataManager** can also retrieve and consume OData v4 services. For more details on OData v4 Services, refer the [odata documentation](#). You can use the **ODataV4Adaptor** to interact with ODataV4 service.

APP.JSX

```
import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { Row } from './rowTemplate';
const SERVICE_URI =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = { items: [] };
  }
  componentDidMount() {
    new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor })
      .executeQuery(new Query().take(8))
      .then((e) => {
        const res = e.result.map((row, index) => (
          <Row key={row.OrderID} {...row} />
        ));
        this.setState({
          items: res
        });
      });
  }
  render() {
    return (
      <table id='datatable' className='e-table'>
        <thead>
          <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
        </thead>
        <tbody>{getValue('items', this.state)}</tbody>
      </table>);
  }
}
```

APP.TSX

```
import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query, ReturnOption } from
 '@syncfusion/ej2-data';
import * as React from 'react';
import { IOrders } from './orders';
import { Row } from './rowTemplate';
const SERVICE_URI: string =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component<{}, {}>{
  constructor(props: object) {
    super(props);
    this.state = { items: [] };
  }
}
```

```

componentDidMount() {
    new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor })
        .executeQuery(new Query().take(8))
        .then((e: ReturnOption) => {
            const res = (e.result as IOOrders[]).map((row: IOOrders) => (
                <Row key={row.OrderID} {...row} />
            ));
            this.setState({
                items: res
            });
        });
}

public render() {
    return (<table id='datatable' className='e-table'>
        <thead>
            <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
        </thead>
        <tbody>{ getValue('items', this.state) }</tbody>
    </table>)
}
}

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```

export interface IOOrders {
    OrderID: number;
    EmployeeID: number;
    CustomerID: string;
    Order_Details: object[];
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
export class Row extends React.Component {
    render() {
        const item = this.props;
        return (<tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>);
    }
}

```

ROWTEMPLATE.TSX

```

import * as React from 'react';
import { IOOrders } from './orders';
export class Row extends React.Component<{}, {}>{

```



```

public render() {
    const item: IOrders = this.props as IOrders;
    return (<tr>
        <td>{item.OrderID}</td>
        <td>{item.CustomerID}</td>
        <td>{item.EmployeeID}</td>
    </tr>)
}
}

```

Web API adaptor

You can use the **WebApiAdaptor** to interact with Web API created with OData endpoint. The **WebApiAdaptor** is extended from the **ODataAdaptor**. Hence to use **WebApiAdaptor**, the endpoint should understand the OData formatted queries send along with request.

To enable OData query option for Web API, please refer to the [documentation](#)

`ts

```

import { DataManager, Query, ReturnOption, WebApiAdaptor } from '@syncfusion/ej2-data';
const SERVICE_URI: string = 'https://ej2services.syncfusion.com/react/development/api/Orders';
new DataManager({
    adaptor: new WebApiAdaptor,
    url: SERVICE_URI
}).executeQuery(new Query().take(8)).then((e: ReturnOption) => {
    // e.result will contain the records
});

```

WebApiAdaptor expects JSON response from the server and the response object should contain properties **Items** and **Count** whose values are collection of entities and total count of the entities respectively.

The sample response object should look like below.

```

{
    Items: [{..}, {..}, {..}, ...],
    Count: 830
}

```

WebMethod Adaptor

The **WebMethodAdaptor** is used to bind data source from remote services and code behind methods. It can be enabled in Grid using Adaptor property of DataManager as **WebMethodAdaptor**.

For every operations, an Fetch post will be send to the specified data service.

```
`ts
import { DataManager, Query, ReturnOption, WebMethodAdaptor } from '@syncfusion/ej2-data';
const SERVICE_URI: string = 'http://controller.com/api';
new DataManager({
  adaptor: new WebMethodAdaptor,
  url: SERVICE_URI
}).executeQuery(new Query().take(8)).then((e: ReturnOption) => {
  // e.result will contain the records
});
`
```

WebMethodAdaptor expects JSON response from the server and the response object should contain properties **result** and **count** whose values are collection of entities and total count of the entities respectively.

The sample response object should look like below.

```
`
{
  result: [{..}, {..}, {..}, ...],
  count: 830
}
`
```

The controller method's data parameter name must be **value**.

GraphQL Adaptor

The **GraphQLAdaptor** provides an option to retrieve data from the GraphQL server. It performs CRUD and data operations such as paging, sorting, filtering etc by sending the required arguments to the server.

You can provide the GraphQL query string by using the **query** property of the **GraphQLAdaptor**. Since, the **GraphQLAdaptor** is extended from the **UrlAdaptor**, it expects response as a JSON object with properties **result** and **count** which contains the collection of entities and the total number of records respectively. The GraphQL response should be returned in JSON format like { "data": { ... } } with query name as field, you need to set the **result** and **count** properties to map the response.

```
`ts
import { DataManager, Query, GraphQLAdaptor } from '@syncfusion/ej2-data';
const SERVICE_URI: string = 'http://controller.com/actions';
new DataManager({
  url: SERVICE_URI, adaptor: new GraphQLAdaptor({
```

```
response: {  
  result: 'getOrders.OrderData',  
  count: 'getOrders.OrderCount'  
},  
query: `query getOrders($datamanager: String) {  
  getOrders(datamanager: $datamanager) {  
    OrderCount,  
    OrderData{OrderID, CustomerID, EmployeeID, ShipCity, ShipCountry}  
  }  
}`  
})  
}).executeQuery(new Query().take(8)).then((e) => {  
  //e.result will contain the records  
});  
`
```

The Schema for the GraphQL server is

```
`ts  
input OrderInput {  
  OrderID: Int!  
  CustomerID: String!  
  EmployeeID: Int!  
  ShipCity: String!  
  ShipCountry: String!  
}  
type Order {  
  OrderID: Int!  
  CustomerID: String!  
  EmployeeID: Int!  
  ShipCity: String!  
  ShipCountry: String!  
}  
type Returntype {  
  getOrders: [Order]
```

```

count: Int
}
type Query {
  getOrders(datamanager: String): ReturnType
}
type Mutation {
  createOrder(value: OrderInput): Order!
  updateOrder(key: Int!, keyColumn: String, value: OrderInput): Order!
  deleteOrder(key: Int!, keyColumn: String, value: OrderInput): Order!
}
`

```

The resolver for the corresponding action is

```

`ts
import { data } from "./db";
const resolvers = {
  Query: {
    getOrders: (parent, { datamanager }, context, info) => {
      if (datamanager.search) {
        // Perform searching
      }
      if (datamanager.sorted) {
        // Perform sorting
      }
      if (datamanager.where) {
        // Perform filtering
      }
      if (datamanager.search) {
        // Perform search
      }
      if (datamanager.skip && datamanager.take) {
        // Perform Paging
      }
      return { OrderData: data, OrderCount: data.length };
    }
  }
}

```

```

}
},
Mutation: {
  createOrder: (parent, { value }, context, info) => {
    // Perform Insert
    return value;
  },
  updateOrder: (parent, { key, keyColumn, value }, context, info) => {
    // Perform Update
    return value;
  },
  deleteOrder: (parent, { key, keyColumn, value }, context, info) => {
    // Perform Delete
    return value;
  },
};
export default resolvers;
`

```

The query parameters will be send in a string format which contains the below details.

Parameters	Description
RequiresCounts	If it is true then the total count of records will be included in response.
Skip	Holds the number of records to skip.
Take	Holds the number of records to take.
Sorted	Contains details about current sorted column and its direction.
Where	Contains details about current filter column name and its constraints.
Group	Contains details about current Grouped column names.

Performing CRUD action with GraphQLAdaptor

You can perform the CRUD actions by returning the mutation queries inside the **getMutation** method based on the action.

```

`ts
import { DataManager, Query, GraphQLAdaptor } from '@syncfusion/ej2-data';

```

```
const SERVICE_URI: string = 'http://controller.com/actions';
new DataManager({
  url: SERVICE_URI, adaptor: new GraphQLAdaptor({
    response: {
      result: 'getOrders.getOrders',
      count: 'getOrders.count'
    },
    query: `query getOrders($datamanager: String) {
      getOrders(datamanager: $datamanager) {
        count,
        getOrders{OrderID, CustomerID, EmployeeID, ShipCity, ShipCountry}
      }
    }`,
    getMutation: function (action): string {
      if (action === 'insert') {
        return `mutation CreateOrderMutation($value: OrderInput!){
          createOrder(value: $value){
            OrderID, CustomerID, EmployeeID, ShipCity, ShipCountry
          }
        }`;
      }
      if (action === 'update') {
        return `mutation Update($key: ID!, $keyColumn: String,$value: OrderInput){
          updateOrder(key: $key, keyColumn: $keyColumn, value: $value) {
            OrderID, CustomerID, EmployeeID, ShipCity, ShipCountry
          }
        }`;
      }
      if (action === 'delete') {
        return `mutation Remove($key: ID!, $keyColumn: String, $value: OrderInput){
          deleteOrder(key: $key, keyColumn: $keyColumn, value: $value) {
            OrderID, CustomerID, EmployeeID, ShipCity, ShipCountry
          }
        }`;
      }
    }
  })
}
```

```

}
})
}).executeQuery(new Query().take(8)).then((e) => {
//e.result will contain the records
});
`

```

Writing custom adaptor

Sometimes the built-in adaptors does not meet your requirement. In such cases you can create your own adaptor.

To create and use custom adaptor, please refer to the below steps.

- Select an built-in adaptor which will act as base class for your custom adaptor.
- Override the desired method to achieve your requirement.
- Assign the custom adaptor to the `adaptor` property of `DataManager`.

For the sake of demonstrating custom adaptor approach, we are going to see how to add serial number for the records by overriding the built-in response processing using `processResponse` method of the `ODataV4Adaptor`.

APP.JSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { Row } from './rowTemplate';
import { SerialNoAdaptor } from './serialNoAdaptor';
const SERVICE_URI =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = { items: [] };
  }
  componentDidMount() {
    new DataManager({ url: SERVICE_URI, adaptor: new SerialNoAdaptor })
      .executeQuery(new Query().take(8))
      .then((e) => {
        const res = e.result.map((row) => (
          <Row key={row.OrderID} {...row} />
        ));
        this.setState({
          items: res,
        });
      });
  }
  render() {
    return (
      <table id='datatable' className='e-table'>
        <thead>
          <tr><th>SNO</th><th>Customer ID</th><th>Employee
ID</th></tr>

```

```

        </thead>
        <tbody>{getValue('items', this.state)}</tbody>
    </table>);
    }
}

```

APP.TSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, Query, ReturnOption } from '@syncfusion/ej2-data';
import * as React from 'react';
import { IOrders } from './orders';
import { Row } from './rowTemplate';
import { SerialNoAdaptor } from './serialNoAdaptor';
const SERVICE_URI: string =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component<{}, {}>{
    constructor(props: object) {
        super(props);
        this.state = { items: [] };
    }
    componentDidMount() {
        new DataManager({ url: SERVICE_URI, adaptor: new SerialNoAdaptor })
            .executeQuery(new Query().take(8))
            .then((e: ReturnOption) => {
                const res = (e.result as IOrders[]).map((row: IOrders) => (
                    <Row key={row.OrderID} {...row} />
                ));
                this.setState({
                    items: res,
                });
            });
    }
    public render() {
        return (<table id='datatable' className='e-table'>
            <thead>
                <tr><th>SNO</th><th>Customer ID</th><th>Employee
ID</th></tr>
            </thead>
            <tbody>{ getValue('items', this.state) }</tbody>
        </table>)
    }
}

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```

export interface IOrders {
    SNO: number,
    OrderID: number;
    EmployeeID: number;
}

```



```

    CustomerID: string;
    Order_Details: object[];
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
export class Row extends React.Component {
    render() {
        const item = this.props;
        return (<tr>
            <td>{item.SNO}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>);
    }
}

```

ROWTEMPLATE.TSX

```

import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
    public render() {
        const item: IOrders = this.props as IOrders;
        return (<tr>
            <td>{item.SNO}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>);
    }
}

```

SERIALNOADAPTOR.JSX

```

import { setValue } from '@syncfusion/ej2-base';
import { ODataV4Adaptor } from '@syncfusion/ej2-data';
export class SerialNoAdaptor extends ODataV4Adaptor {
    processResponse() {
        let i = 0;
        // calling base class processResponse function
        const original = super.processResponse.apply(this, arguments);
        // adding serial number
        original.forEach((item) => setValue('SNO', ++i, item));
        return original;
    }
}

```

SERIALNOADAPTOR.TSX

```

import { setValue } from '@syncfusion/ej2-base';
import { ODataV4Adaptor } from '@syncfusion/ej2-data';
export class SerialNoAdaptor extends ODataV4Adaptor {
    public processResponse() {
        let i: number = 0;

```

```

        // calling base class processResponse function
        const original: any = super.processResponse.apply(this, arguments as
any);

        // adding serial number
        original.forEach((item: object) => setValue('SNO', ++i, item));
        return original;
    }
}

```

Querying in React Data component

In this section, you will see in detail about how to build query using [Query](#) class and consume the data source.

Specifying resource name using `from`

The `from` method is used to specify the resource name or table name from where the data should be retrieved.

APP.JSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { Row } from './rowTemplate';
const SERVICE_URI = 'https://services.odata.org/V4/Northwind/Northwind.svc/';
export default class App extends React.Component {
    constructor(props) {
        super(props);
        this.state = { items: [] };
    }
    componentDidMount() {
        new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
            .executeQuery(new Query().from('Orders').take(8))
            .then((e) => {
                const res = e.result.map((row) => (
                    <Row key={row.OrderID} {...row} />
                ));
                this.setState({
                    items: res
                });
            });
    }
    render() {
        return (
            <table id='datatable' className='e-table'>
                <thead>
                    <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
                </thead>
                <tbody>{getValue('items', this.state)}</tbody>
            </table>);
    }
}

```

APP.TSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query, ReturnOption } from
 '@syncfusion/ej2-data';
import * as React from 'react';
import { IOrders } from './orders';
import { Row } from './rowTemplate';
const SERVICE_URI:string=
 'https://services.odata.org/V4/Northwind/Northwind.svc/';
export default class App extends React.Component<{}, {}>{
  constructor(props: object) {
    super(props);
    this.state = { items: [] };
  }
  componentDidMount() {
    new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
    .executeQuery(new Query().from('Orders').take(8))
    .then((e: ReturnOption) => {
      const res = (e.result as IOrders[]).map((row: IOrders, index:
number) => (
        <Row key={row.OrderID} {...row} />
      ));
      this.setState({
        items: res
      });
    });
  }
  public render() {
    return <table id='datatable' className='e-table'>
      <thead>
        <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
      </thead>
      <tbody>{ getValue('items', this.state) }</tbody>
    </table>
  }
}

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```

export interface IOrders {
  OrderID: number;
  EmployeeID: number;
  CustomerID: string;
  Order_Details: object[];
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
export class Row extends React.Component {
  render() {

```

```

    const item = this.props;
    return (<tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>);
  }
}

```

ROWTEMPLATE.TSX

```

import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
  public render() {
    const item: IOrders = this.props as IOrders;
    return (<tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>);
  }
}

```

Projection using select

The [select](#) method is used to select particular fields or columns from the data source.

APP.JSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { Row } from './rowTemplate';
const SERVICE_URI =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders';
export default class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = { items: [] };
  }
  componentDidMount() {
    new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
      .executeQuery(new Query().select(['OrderID',
'CustomerID', 'EmployeeID']).take(8))
      .then((e) => {
        const res = e.result.map((row) => (
          <Row key={row.OrderID} {...row} />
        ));
        this.setState({
          items: res
        });
      });
  }
  render() {
    return (<table id='datatable' className='e-table'>
      <thead>

```

```

        <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
        </thead>
        <tbody>{getValue('items', this.state)}</tbody>
    </table>);
    }
}

```

APP.TSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query, ReturnOption } from
 '@syncfusion/ej2-data';
import * as React from 'react';
import { IOrders } from './orders';
import { Row } from './rowTemplate';
const SERVICE_URI:string =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders';
export default class App extends React.Component<{}, {}>{
    constructor(props: object) {
        super(props);
        this.state = { items: [] };
    }
    componentDidMount() {
        new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
        .executeQuery(new Query().select(['OrderID',
'CustomerID', 'EmployeeID']).take(8))
        .then((e: ReturnOption) => {
            const res = (e.result as IOrders[]).map((row: IOrders, index:
number) => (
                <Row key={row.OrderID} {...row} />
            ));
            this.setState({
                items: res
            });
        });
    }
    public render() {
        return (<table id='datatable' className='e-table'>
            <thead>
                <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
            </thead>
            <tbody>{ getValue('items', this.state) }</tbody>
        </table>)
    }
}

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```
export interface IOrders {
```

```

    OrderID: number;
    EmployeeID: number;
    CustomerID: string;
    Order_Details: object[];
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
export class Row extends React.Component {
    render() {
        const item = this.props;
        return (
            <tr>
                <td>{item.OrderID}</td>
                <td>{item.CustomerID}</td>
                <td>{item.EmployeeID}</td>
            </tr>
        );
    }
}

```

ROWTEMPLATE.TSX

```

import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
    public render() {
        const item: IOrders = this.props as IOrders;
        return (
            <tr>
                <td>{item.OrderID}</td>
                <td>{item.CustomerID}</td>
                <td>{item.EmployeeID}</td>
            </tr>
        );
    }
}

```

Eager loading navigation properties

You can use the [expand](#) method to eagerly load navigation properties. The navigation properties values are accessed using appropriate field names separated by dot(.) sign.

APP.JSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { Row } from './rowTemplate';
const SERVICE_URI =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component {
    constructor(props) {
        super(props);
        this.state = { items: [] };
    }
    componentDidMount() {
        new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })

```

```

        .executeQuery(new Query()
            .expand('Employee')
            .select(['OrderID', 'CustomerID', 'Employee.FirstName'])
            .take(8))
        .then((e) => {
            const res = e.result.map((row) => <Row key={row.OrderID}
{...row} />);
            this.setState({
                items: res,
            });
        });
    }
    render() {
        return (<table id='datatable' className='e-table'>
            <thead>
                <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
            </thead>
            <tbody>{getValue('items', this.state)}</tbody>
        </table>);
    }
}

```

APP.TSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query, ReturnOption } from
 '@syncfusion/ej2-data';
import * as React from 'react';
import { IOOrders } from './orders';
import { Row } from './rowTemplate';
const SERVICE_URI: string =
 'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component<{}, {}>{
    constructor(props: object) {
        super(props);
        this.state = { items: [] };
    }
    componentDidMount() {
        new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
            .executeQuery(
                new Query()
                    .expand('Employee')
                    .select(['OrderID', 'CustomerID', 'Employee.FirstName'])
                    .take(8)
            )
            .then((e: ReturnOption) => {
                const res = (e.result as IOOrders[]).map((row: IOOrders) => (
                    <Row key={row.OrderID} {...row} />
                ));
                this.setState({
                    items: res
                });
            });
    }
    public render() {

```

```

        return (<table id='datatable' className='e-table'>
            <thead>
                <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
            </thead>
            <tbody>{ getValue('items', this.state) }</tbody>
        </table>)
    }
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
;
export class Row extends React.Component {
    render() {
        const item = this.props;
        return (<tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.Employee.FirstName}</td>
        </tr>);
    }
}

```

ROWTEMPLATE.TSX

```

import * as React from 'react';
interface IOrders {OrderID: number, CustomerID: string, Employee: {FirstName:
string} };
export class Row extends React.Component<{}, {}>{
    public render() {
        const item: IOrders = this.props as IOrders;
        return (<tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.Employee.FirstName}</td>
        </tr>)
    }
}

```

Sorting

You can use the [sortBy](#) method to perform sort operation in the data source. Default sorting order is **ascending**. To change the sort order, either you can specify the second argument of [sortBy](#) as **descending** or use the [sortByDesc](#) method.

APP.JSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { Row } from './rowTemplate';
const SERVICE_URI =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders';
export default class App extends React.Component {

```



```

    constructor(props) {
        super(props);
        this.state = { items: [] };
    }
    componentDidMount() {
        new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
            .executeQuery(new Query().sortBy('CustomerID',
'descending').take(8))
            .then((e) => {
                const res = e.result.map((row) => (
                    <Row key={row.OrderID} {...row} />
                ));
                this.setState({
                    items: res
                });
            });
    }
    render() {
        return <table id='datatable' className='e-table'>
            <thead>
                <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
            </thead>
            <tbody>{getValue('items', this.state)}</tbody>
        </table>;
    }
}

```

APP.TSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query, ReturnOption } from
 '@syncfusion/ej2-data';
import * as React from 'react';
import { IOOrders } from './orders';
import { Row } from './rowTemplate';
const SERVICE_URI:string=
 'https://services.odata.org/V4/Northwind/Northwind.svc/Orders';
export default class App extends React.Component<{}, {}>{
    constructor(props: object) {
        super(props);
        this.state = { items: [] };
    }
    componentDidMount() {
        new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
            .executeQuery(new Query().sortBy('CustomerID',
'descending').take(8))
            .then((e: ReturnOption) => {
                const res = (e.result as IOOrders[]).map((row: IOOrders,index:
number) => (
                    <Row key={row.OrderID} {...row} />
                ));
                this.setState({
                    items: res
                });
            });
    }
}

```

```

    }
    public render() {
        return (<table id='datatable' className='e-table'>
            <thead>
                <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
            </thead>
            <tbody>{ getValue('items', this.state) }</tbody>
        </table>)
    }
}

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```

export interface IOrders {
    OrderID: number;
    EmployeeID: number;
    CustomerID: string;
    Order_Details: object[];
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
export class Row extends React.Component {
    render() {
        const item = this.props;
        return (<tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>);
    }
}

```

ROWTEMPLATE.TSX

```

import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
    public render() {
        const item: IOrders = this.props as IOrders;
        return (<tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>)
    }
}

```

Multi sorting can be performed by simply chaining the multiple `sortBy` methods.

Filtering

You can use the [where](#) method to build filter criteria which allows you to get reduced view of records. The [where](#) method can also be chained to form multiple filter criteria.

APP.JSX

```
import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { Row } from './rowTemplate';
const SERVICE_URI =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders';
export default class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = { items: [] };
  }
  componentDidMount() {
    new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
      .executeQuery(new Query().where('EmployeeID', 'equal',
3).take(8))
      .then((e) => {
        const res = e.result.map((row) => (
          <Row key={row.OrderID} {...row} />
        ));
        this.setState({
          items: res
        });
      });
  }
  render() {
    return (
      <table id='datatable' className='e-table'>
        <thead>
          <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
        </thead>
        <tbody>{getValue('items', this.state)}</tbody>
      </table>);
  }
}
```

APP.TSX

```
import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query, ReturnOption } from
 '@syncfusion/ej2-data';
import * as React from 'react';
import { IOrders } from './orders';
import { Row } from './rowTemplate';
const SERVICE_URI:string=
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders';
export default class App extends React.Component<{}, {}>{
  constructor(props: object) {
    super(props);
  }
```

```

        this.state = { items: [] };
    }
    componentDidMount() {
        new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
            .executeQuery(new Query().where('EmployeeID', 'equal',
3).take(8))
            .then((e: ReturnOption) => {
                const res = (e.result as IOOrders[]).map((row: IOOrders, index:
number) => (
                    <Row key={row.OrderID} {...row} />
                ));
                this.setState({
                    items: res
                });
            });
    }
    public render() {
        return <table id='datatable' className='e-table'>
            <thead>
                <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
            </thead>
            <tbody>{ getValue('items', this.state) }</tbody>
        </table>
    }
}

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```

export interface IOOrders {
    OrderID: number;
    EmployeeID: number;
    CustomerID: string;
    Order_Details: object[];
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
export class Row extends React.Component {
    render() {
        const item = this.props;
        return <tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>;
    }
}

```

ROWTEMPLATE.TSX

```
import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
  public render() {
    const item: IOrders = this.props as IOrders;
    return (<tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>)
  }
}
```

Filter Operators

Filter operators are generally used to specify the filter type. The various filter operators supported by [DataManager](#) is listed below.

- greaterthan
- greaterthanorequal
- lessthan
- lessthanorequal
- equal
- notequal
- startswith
- endswith
- contains

These filter operators are used for creating filter query using [where](#) method and [Predicate](#) class.

Build complex filter criteria using `Predicate`

Sometimes chaining `where` method is not sufficient to create very complex filter criteria, in such cases we can use [Predicate](#) class to create composite filter criteria.

APP.JSX

```
import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Predicate, Query } from
 '@syncfusion/ej2-data';
import * as React from 'react';
import { Row } from './rowTemplate';
const SERVICE_URI =
 'https://services.odata.org/V4/Northwind/Northwind.svc/Orders';
export default class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = { items: [] };
  }
  componentDidMount() {
    let predicate = new Predicate('EmployeeID', 'equal', 3);
    predicate = predicate.or('EmployeeID', 'equal', 2);

    new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
```

```

        .executeQuery(new Query().where(predicate).take(8))
        .then((e) => {
            const res = e.result.map((row) => (
                <Row key={row.OrderID} {...row} />
            ));
            this.setState({
                items: res
            });
        });
    }
    render() {
        return (<table id='datatable' className='e-table'>
            <thead>
                <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
            </thead>
            <tbody>{getValue('items', this.state)}</tbody>
        </table>);
    }
}

```

APP.TSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Predicate, Query, ReturnOption } from
 '@syncfusion/ej2-data';
import * as React from 'react';
import { IOOrders } from './orders';
import { Row } from './rowTemplate';
const SERVICE_URI:string =
 'https://services.odata.org/V4/Northwind/Northwind.svc/Orders';
export default class App extends React.Component<{}, {}>{
    constructor(props: object) {
        super(props);
        this.state = { items: [] };
    }
    componentDidMount() {
        let predicate: Predicate = new Predicate('EmployeeID', 'equal', 3);
        predicate = predicate.or('EmployeeID', 'equal', 2);
        new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
            .executeQuery(new Query().where(predicate).take(8))
            .then((e: ReturnOption) => {
                const res = (e.result as IOOrders[]).map((row: IOOrders, index:
number) => (
                    <Row key={row.OrderID} {...row} />
                ));
                this.setState({
                    items: res
                });
            });
    }
    public render() {
        return (<table id='datatable' className='e-table'>
            <thead>
                <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>

```

```

        </thead>
        <tbody>{ getValue('items', this.state) }</tbody>
    </table>)
    }
}

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```

export interface IOrders {
    OrderID: number;
    EmployeeID: number;
    CustomerID: string;
    Order_Details: object[];
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
export class Row extends React.Component {
    render() {
        const item = this.props;
        return <tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>;
    }
}

```

ROWTEMPLATE.TSX

```

import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
    public render() {
        const item: IOrders = this.props as IOrders;
        return <tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>
    }
}

```

Searching

You can use the [search](#) method to create search criteria, it differs from the filter in the way that search criteria will be applied to all fields in the datasource whereas filter criteria will be applied to a particular field.

APP.JSX

```
import { DataManager , Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { Row } from './rowTemplate';
export default class App extends React.Component {
    result = new DataManager({ json: data })
        .executeLocal(new Query().search('VI', ['CustomerID']));
    items = this.result.map((row, index) => (
        <Row key={index} {...row} />
    ));
    render() {
        return (<table id='datatable' className='e-table'>
            <thead>
                <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
            </thead>
            <tbody>{this.items}</tbody>
        </table>);
    }
}
```

APP.TSX

```
import { DataManager, Query } from '@syncfusion/ej2-data';
import { data } from './datasource';
import { Row } from './rowTemplate';
import * as React from 'react';
export default class App extends React.Component<{}, {}> {
    public result: Object[] = new DataManager({ json: data })
        .executeLocal(new Query().search('VI', ['CustomerID']));

    public items: React.ReactElement[] = this.result.map((row: object, index)
=> (
        <Row key={index} {...row} />
    ));
    public render() {
        return (
            <table id='datatable' className='e-table'>
                <thead>
                    <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
                </thead>
                <tbody>{this.items}</tbody>
            </table>
        );
    }
}
```

ORDERS.JSX

```
export {};
```

ORDERS.TSX


```
export interface IOrders {
  OrderID: number;
  EmployeeID: number;
  CustomerID: string;
  Order_Details: object[];
}
```

ROWTEMPLATE.JSX

```
import * as React from 'react';
export class Row extends React.Component {
  render() {
    const item = this.props;
    return (<tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>);
  }
}
```

ROWTEMPLATE.TSX

```
import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
  public render() {
    const item: IOrders = this.props as IOrders;
    return (<tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>);
  }
}
```

You can search particular fields by passing the field name collection in the second argument of [search](#) method.

Grouping

[DataManager](#) allow you to group records by category. The [group](#) method is used to add group query.

APP.JSX

```
import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { GroupRow } from './groupTemplate';
const SERVICE_URI =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = { items: [] };
  }
}
```

```

componentDidMount() {
    new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
        .executeQuery(new Query().group('CustomerID').take(20))
        .then((e) => {
            const res = e.result.map((row, index) => (
                <GroupRow key={index} {...row} />
            ));
            this.setState({
                items: res
            });
        });
}
render() {
    return (
        <table id='datatable' className='e-table'>
            <thead>
                <tr>
                    <th>Order ID</th>
                    <th>Customer ID</th>
                    <th>Employee ID</th>
                </tr>
            </thead>
            {getValue('items', this.state)}
        </table>
    );
}
}

```

APP.TSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, Group, ODataV4Adaptor, Query, ReturnOption } from
 '@syncfusion/ej2-data';
import * as React from 'react';
import { GroupRow } from './groupTemplate';
const SERVICE_URI: string =
 'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component<{}, {}> {
    constructor(props: object) {
        super(props);
        this.state = { items: [] };
    }
    componentDidMount() {
        new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
            .executeQuery(new Query().group('CustomerID').take(20))
            .then((e: ReturnOption) => {
                const res = (e.result as Group[]).map((row: Group,
index:number) => (
                    <GroupRow key={index} {...row} />
                ));
                this.setState({
                    items: res,
                });
            });
    }
    public render() {

```

```

        return (
            <table id='datatable' className='e-table'>
                <thead>
                    <tr>
                        <th>Order ID</th>
                        <th>Customer ID</th>
                        <th>Employee ID</th>
                    </tr>
                </thead>
                {getValue('items', this.state)}
            </table>
        );
    }
}

```

GROUPTEMPLATE.JSX

```

import * as React from 'react';
import { Row } from './rowTemplate';
export class GroupRow extends React.Component {
    getRows(data) {
        return data.map((row, index) => (
            <Row key={index} {...row} />
        ));
    }
    render() {
        const item = this.props;
        const ag = { caption: item.field + ' - ' + (item.items &&
            item.items[0][item.field]) };
        return (
            <tbody>
                <Row key={ag.caption} {...ag} />
                {this.getRows(item.items)}
            </tbody>
        );
    }
}

```

GROUPTEMPLATE.TSX

```

import { Group } from '@syncfusion/ej2-data';
import * as React from 'react';
import { IOOrders } from './orders';
import { Row } from './rowTemplate';
export class GroupRow extends React.Component<{}, {}> {
    public getRows(data: IOOrders[]) {
        return data.map((row: IOOrders, index: number) => (
            <Row key={index} {...row} />
        ));
    }
    public render() {
        const item: Group = this.props as Group;
        const ag: { caption: string } = {
            caption: item.field + ' - ' + (item.items && (item.items[0] as
            Object[][item.field as any])),
        };
    }
}

```

```

        return (
            <tbody>
                <Row key={ag.caption} {...ag} />
                {this.getRows(item.items as IOrders[])}
            </tbody>
        );
    }
}

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```

export interface IOrders {
    SNO: number,
    OrderID: number;
    EmployeeID: number;
    CustomerID: string;
    // Order_Details: object[];
}

```

ROWTEMPLATE.JSX

```

import { getValue } from '@syncfusion/ej2-base';
import * as React from 'react';
export class Row extends React.Component {
    render() {
        const item = this.props;
        if (getValue('caption', item)) {
            return <tr>
                <td colSpan={3}>{getValue('caption', item)}</td>
            </tr>;
        }
        return <tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>;
    }
}

```

ROWTEMPLATE.TSX

```

import { getValue } from '@syncfusion/ej2-base';
import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
    public render() {
        const item: IOrders = this.props as IOrders;
        if (getValue('caption', item)) {
            return <tr>
                <td colSpan={3}>{getValue('caption', item)}</td>
            </tr>
        }
    }
}

```

```

    }
    return (<tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>)
  }
}

```

Multiple grouping can be done by simply chaining the `group` method.

Paging

You can query paged data using `page` method. This allow you to query particular set of records based on the page size and index.

APP.JSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Predicate, Query } from
 '@syncfusion/ej2-data';
import * as React from 'react';
import { Row } from './rowTemplate';
const SERVICE_URI =
 'https://services.odata.org/V4/Northwind/Northwind.svc/Orders';
export default class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = { items: [] };
  }
  componentDidMount() {
    new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
      .executeQuery(new Query().page(2, 8))
      .then((e) => {
        const res = e.result.map((row) => (
          <Row key={row.OrderID} {...row} />
        ));
        this.setState({
          items: res
        });
      });
  }
  render() {
    return (<table id='datatable' className='e-table'>
      <thead>
        <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
      </thead>
      <tbody>{getValue('items', this.state)}</tbody>
    </table>);
  }
}

```

APP.TSX

```

import { getValue } from '@syncfusion/ej2-base';

```

```

import { DataManager, ODataV4Adaptor, Query, ReturnOption } from
'@syncfusion/ej2-data';
import * as React from 'react';
import { IOrders } from './orders';
import { Row } from './rowTemplate';
const SERVICE_URI:string =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders';
export default class App extends React.Component<{}, {}>{
  constructor(props: object) {
    super(props);
    this.state = { items: [] };
  }
  componentDidMount() {
    new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
      .executeQuery(new Query().page(2, 8))
      .then((e: ReturnOption) => {
        const res = (e.result as IOrders[]).map((row: IOrders, index:
number) => (
          <Row key={row.OrderID} {...row} />
        ));
        this.setState({
          items: res
        });
      });
  }
  public render() {
    return <table id='datatable' className='e-table'>
      <thead>
        <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
      </thead>
      <tbody>{ getValue('items', this.state) }</tbody>
    </table>
  }
}

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```

export interface IOrders {
  OrderID: number;
  EmployeeID: number;
  CustomerID: string;
  Order_Details: object[];
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
export class Row extends React.Component {
  render() {
    const item = this.props;
  }
}

```

```

        return (<tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>);
    }
}

```

ROWTEMPLATE.TSX

```

import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
    public render() {
        const item: IOrders = this.props as IOrders;
        return (<tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>)
    }
}

```

Aggregation

The [aggregate](#) method allows you to get aggregated value for a field based on the type.

The built-in aggregate types are,

- sum
- average
- min
- max
- count
- truecount
- falsecount

AGGREGATETEMPLATE.JSX

```

import * as React from 'react';
export class AggregateRow extends React.Component {
    render() {
        const item = this.props;
        return (<tr>
            <td />
            <td />
            <td>Min: {item.min}</td>
        </tr>);
    }
}

```

AGGREGATETEMPLATE.TSX

```

import { Aggregates } from '@syncfusion/ej2-data';

```

```
import * as React from 'react';
export class AggregateRow extends React.Component<{ }, {}> {
  public render() {
    const item: Aggregates = this.props as Aggregates;
    return (
      <tr>
        <td/>
        <td/>
        <td>{`Min: ${item.min}`}</td>
      </tr>
    );
  }
}
```

APP.JSX

```
import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { AggregateRow } from './aggregateTemplate';
import { Row } from './rowTemplate';
const SERVICE_URI =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = { items: [], aggregates: [] };
  }
  componentDidMount() {
    new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
      .executeQuery(new
Query().take(5).requiresCount().aggregate('min', 'EmployeeID'))
      .then((e) => {
        const agg = { min: e.aggregates['EmployeeID - min'] };
        const ret = <AggregateRow key="aggregate" {...agg} />;
        const res = e.result.map((row) => (
          <Row key={row.OrderID} {...row} />
        ));
        this.setState({
          aggregates: [ret],
          items: res
        });
      });
  }
  render() {
    return (<table id='datatable' className='e-table'>
      <thead>
        <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
      </thead>
      <tbody>{getValue('items', this.state)}</tbody>
      <tfoot>{getValue('aggregates', this.state)}</tfoot>
    </table>);
  }
}
```


APP.TSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query, ReturnOption } from
 '@syncfusion/ej2-data';
import * as React from 'react';
import { AggregateRow } from './aggregateTemplate';
import { IOrders } from './orders';
import { Row } from './rowTemplate';
const SERVICE_URI: string =
 'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component<{}, {}> {
  constructor(props: object) {
    super(props);
    this.state = { items: [], aggregates: [] };
  }
  componentDidMount() {
    new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
      .executeQuery(new Query().requiresCount().take(5).aggregate('min',
 'EmployeeID'))
      .then((e: ReturnOption) => {
        console.log(e.aggregates)
        const agg: { min: object } = { min: (e.aggregates as
 AggregatesType) ['EmployeeID - min'] };
        const ret: React.ReactElement = <AggregateRow key="aggregate"
 {...agg} />;
        const res = ((e.result as IOrders[])).map((row: IOrders, index:
 number) => (
          <Row key={row.OrderID} {...row} />
        ));
        this.setState({
          aggregates: [ret],
          items: res,
        });
      });
  }
  public render() {
    return (
      <table id="datatable" className="e-table">
        <thead>
          <tr>
            <th>Order ID</th>
            <th>Customer ID</th>
            <th>Employee ID</th>
          </tr>
        </thead>
        <tbody>{getValue('items', this.state)}</tbody>
        <tfoot>{getValue('aggregates', this.state)}</tfoot>
      </table>
    );
  }
}
interface AggregatesType
{
  [key: string]: object;
};

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```
export interface IOrders {
  OrderID: number;
  EmployeeID: number;
  CustomerID: string;
  Order_Details: object[];
}
```

ROWTEMPLATE.JSX

```
import * as React from 'react';
export class Row extends React.Component {
  render() {
    const item = this.props;
    return (<tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>);
  }
}
```

ROWTEMPLATE.TSX

```
import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
  public render() {
    const item: IOrders = this.props as IOrders;
    return (<tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>);
  }
}
```

Hierarchical query

You can use the [hierarchy](#) method to build nested query. The hierarchical queries are commonly required when you use foreign key binding.

The [foreignKey](#) method is used to specify the key field of the foreign table and the second argument of the [hierarchy](#) method accepts a selector function which selects the records from the foreign table.

APP.JSX

```
import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { Row } from './rowTemplate';
```

```

const SERVICE_URI = 'https://services.odata.org/V4/Northwind/Northwind.svc/';
export default class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = { items: [], aggregates: [] };
  }
  componentDidMount() {
    new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor })
      .executeQuery(new Query().from('Orders').take(3).hierarchy(new
Query()
  .foreignKey("OrderID")
  .from("Order_Details")
  .sortBy("Quantity"), () => [10248, 10249, 10250] // Selective
loading of child elements
))
    .then((e) => {
      const res = e.result.map((row, index) => (<Row key={index}
{...row}/>));
      this.setState({
        items: res
      });
    });
  }
  render() {
    return (<table id='datatable' className='e-table'>
      <thead>
        <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
      </thead>
      {getValue('items', this.state)}
    </table>);
  }
}

```

APP.TSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query, ReturnOption } from
 '@syncfusion/ej2-data';
import * as React from 'react';
import { IOrders } from './orders';
import { Row } from './rowTemplate';
const SERVICE_URI: string =
 'https://services.odata.org/V4/Northwind/Northwind.svc/';
export default class App extends React.Component<{}, {}>{
  constructor(props: object) {
    super(props);
    this.state = { items: [], aggregates: [] };
  }
  componentDidMount() {
    new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
      .executeQuery(new Query().from('Orders').take(3).hierarchy(
        new Query()
          .foreignKey("OrderID")
          .from("Order_Details")
          .sortBy("Quantity"),

```

```

        () => [10248, 10249, 10250] // Selective loading of child
        elements
    ))
    .then((e: ReturnOption) => {
        const res = (e.result as IOOrders[]).map((row:
IOOrders, index: number) => (<Row key={index} {...row}/>));
        this.setState({
            items: res
        });
    });
}

public render() {
    return (<table id='datatable' className='e-table'>
        <thead>
            <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
        </thead>
        {getValue('items', this.state)}
    </table>)
}
}

```

CHILDTEMPLATE.JSX

```

import * as React from 'react';
export class ChildRow extends React.Component {
    render() {
        const item = this.props;
        return (<tr>
            <td>{item.ProductID}</td>
            <td>{item.UnitPrice}</td>
            <td>{item.Quantity}</td>
        </tr>);
    }
}

```

CHILDTEMPLATE.TSX

```

import * as React from 'react';
interface IProducts { ProductID: number; UnitPrice: number; Quantity: number;
}
export class ChildRow extends React.Component<{}, {}>{
    public render() {
        const item: IProducts = this.props as IProducts;
        return (<tr>
            <td>{item.ProductID}</td>
            <td>{item.UnitPrice}</td>
            <td>{item.Quantity}</td>
        </tr>)
    }
}

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```
export interface IOrders {
  OrderID: number;
  EmployeeID: number;
  CustomerID: string;
  Order_Details: object[];
}
```

ROWTEMPLATE.JSX

```
import * as React from 'react';
import { ChildRow } from './childTemplate';
export class Row extends React.Component {
  getRows(data) {
    const dat = data.map((row, index) => (<ChildRow key={index}
{...row}/>));
    return dat;
  }
  render() {
    const item = this.props;
    return (<tbody><tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>
    <tr><td colspan={3}>
      <table id='datatable' className='e-table'>

<thead><tr><th>ID</th><th>Price</th><th>Quantity</th></tr></thead>
      <tbody>{this.getRows(item.Order_Details)}</tbody>
    </table>
    </td></tr>
    </tbody>);
  }
}
```

ROWTEMPLATE.TSX

```
import * as React from 'react';
import { ChildRow } from './childTemplate';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}> {
  public getRows(data: object[] = []) {
    return data.map((row: object, index: number) => (
      <ChildRow key={index} {...row} />
    ));
  }
  public render() {
    const item: IOrders = this.props as IOrders;
    return (
      <tbody>
        <tr>
```

```

        <td>{item.OrderID}</td>
        <td>{item.CustomerID}</td>
        <td>{item.EmployeeID}</td>
      </tr>
      <tr>
        <td colspan={3}>
          <table id='datatable' className='e-table'>
            <thead>
<tr><th>ID</th><th>Price</th><th>Quantity</th></tr>
            </thead>
            <tbody>{this.getRows(item.Order_Details)}</tbody>
          </table>
        </td>
      </tr>
    </tbody>
  );
}
}

```

Manipulation in React Data component

In this section, you will see in detail about how to manipulate data using [DataManager](#). The [DataManager](#) can create, update and delete records either in local data source or remote data source.

Each data sources uses different way in handling the CRUD operations and hence [DataManager](#) uses data adaptors to manipulate data that can be understood by a particular data source.

Insert

The [insert](#) method of [DataManager](#) is used to add new record to the data source. For remote data source, the new record will be send along with the request to the server.

APP.JSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { Row } from './rowTemplate';
export default class App extends React.Component {
  dm;
  style;
  constructor(props) {
    super(props);
    this.state = { items: [] };
    this.style = { class: 'e-form' };
    this.dm = new DataManager(data.slice(0, 5));
    this.dm.executeQuery(new Query())
      .then((e) => {
        this.setState({
          items: e.result.map((row, index) => (
            <Row key={index} {...row} />
          ))
        });
      });
    this.insertUpdate = this.insertUpdate.bind(this);
  }
}

```

```

insertUpdate() {
    const orderid = document.getElementById('OrderID');
    const cusid = document.getElementById('CustomerID');
    const empid = document.getElementById('EmployeeID');
    const rowdata = {
        CustomerID: cusid.value,
        EmployeeID: +empid.value,
        OrderID: +orderid.value
    };
    if (!rowdata.OrderID) {
        return;
    }
    this.dm.insert(rowdata);
    this.dm.executeQuery(new Query())
        .then((e) => {
            this.setState({
                items: e.result.map((row) => (
                    <Row key={row.OrderID} {...row} />
                ))
            });
        });
}

render() {
    return (<div><div style={this.style}>
        <input type="number" id='OrderID' placeholder="Order ID"/>
        <input type="text" id="CustomerID" placeholder="Customer ID"/>
        <input type="number" id="EmployeeID" placeholder="Employee ID"/>
        <input type="button" value="Insert" id="manipulate"
onClick={this.insertUpdate}/></div>
        <div><table id='datatable' className='e-table'>
            <thead>
                <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
            </thead>
            <tbody>{getValue('items', this.state)}</tbody>
        </table>
        </div></div>);
}
}

```

APP.TSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, Query, ReturnOption } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { Row } from './rowTemplate';
export default class App extends React.Component<{}, {}>{
    public dm: DataManager;
    public style: { [x: string]: string };
    constructor(props: object) {
        super(props);
        this.state = { items: [] };
        this.style = { class: 'e-form' };
        this.dm = new DataManager(data.slice(0, 5));
        this.dm.executeQuery(new Query())

```

```

        .then((e: ReturnOption) => {
            this.setState({
                items: (e.result as object[]).map((row: object, index: number) => (
                    <Row key={index} {...row} />
                ))
            });
        });
        this.insertUpdate = this.insertUpdate.bind(this);
    }
    public insertUpdate() {
        const orderid: HTMLInputElement = document.getElementById('OrderID')
as HTMLInputElement;
        const cusid: HTMLInputElement = document.getElementById('CustomerID')
as HTMLInputElement;
        const empid: HTMLInputElement = document.getElementById('EmployeeID')
as HTMLInputElement;
        const rowdata: { OrderID: number, CustomerID: string, EmployeeID:
number } = {
            CustomerID: cusid.value,
            EmployeeID: +empid.value,
            OrderID: +orderid.value
        };
        if (!rowdata.OrderID) { return; }
        this.dm.insert(rowdata);
        this.dm.executeQuery(new Query())
            .then((e: ReturnOption) => {
                this.setState({
                    items: (e.result as object[]).map((row:
object, index: number) => (
                        <Row key={index} {...row} />
                    ))
                });
            });
    }
    public render() {
        return (<div><div style={this.style}>
            <input type="number" id='OrderID' placeholder="Order ID" />
            <input type="text" id="CustomerID" placeholder="Customer ID" />
            <input type="number" id="EmployeeID" placeholder="Employee ID" />
            <input type="button" value="Insert" id="manipulate"
onClick={this.insertUpdate} /></div>
            <div><table id='datatable' className='e-table'>
                <thead>
                    <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
                </thead>
                <tbody>{ getValue('items', this.state) }</tbody>
            </table>
            </div></div>)
    }
}

```

ROWTEMPLATE.JSX

```
import * as React from 'react';
```



```
export class Row extends React.Component {
  render() {
    const item = this.props;
    return (<tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>);
  }
}
```

ROWTEMPLATE.TSX

```
import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
  public render() {
    const item: IOrders = this.props as IOrders;
    return (<tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>);
  }
}
```

In remote data sources, when the primary key field is an identity field, then it is advised to return the created data in the response.

Update

The [update](#) method of [DataManager](#) is used to modify/update a record in the data source. For remote data source, the modified record will be send along with the request to the server.

APP.JSX

```
import { getValue } from '@syncfusion/ej2-base';
import { DataManager, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { Row } from './rowTemplate';
export default class App extends React.Component {
  dm;
  style;
  constructor(props) {
    super(props);
    this.state = { items: [] };
    this.style = { class: 'e-form' };
    this.dm = new DataManager(data.slice(0, 5));
    this.dm.executeQuery(new Query())
      .then((e) => {
        this.setState({
          items: e.result.map((row, index) => (
            <Row key={index} {...row} />
          ))
        });
      });
  }
}
```

```

    });
    this.insertUpdate = this.insertUpdate.bind(this);
  }
  insertUpdate() {
    const orderid = document.getElementById('OrderID');
    const cusid = document.getElementById('CustomerID');
    const empid = document.getElementById('EmployeeID');
    const rowdata = {
      CustomerID: cusid.value,
      EmployeeID: +empid.value,
      OrderID: +orderid.value
    };
    if (!rowdata.OrderID) {
      return;
    }
    this.dm.update('OrderID', rowdata);
    this.dm.executeQuery(new Query())
      .then((e) => {
        this.setState({
          items: e.result.map((row, index) => (
            <Row key={index} {...row} />
          ))
        });
      });
  }
  render() {
    return (<div><div style={this.style}>
      <input type="number" id='OrderID' placeholder="Order ID"/>
      <input type="text" id="CustomerID" placeholder="Customer ID"/>
      <input type="number" id="EmployeeID" placeholder="Employee ID"/>
      <input type="button" value="Update" id="manipulate"
onClick={this.insertUpdate}/></div>
      <div><table id='datatable' className='e-table'>
        <thead>
          <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
        </thead>
        <tbody>{getValue('items', this.state)}</tbody>
      </table>
    </div></div>);
  }
}

```

APP.TSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, Query, ReturnOption } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from '../datasource';
import { Row } from '../rowTemplate';
export default class App extends React.Component<{}, {}>{
  public dm: DataManager;
  public style: { [x: string]: string };
  constructor(props: object) {
    super(props);
    this.state = { items: [] };
  }
}

```

```

        this.style = { class: 'e-form' };
        this.dm = new DataManager(data.slice(0, 5));
        this.dm.executeQuery(new Query())
        .then((e: ReturnOption) => {
            this.setState({
                items: (e.result as object[]).map((row: object, index: number) => (
                    <Row key={index} {...row} />
                ))
            });
        });
        this.insertUpdate = this.insertUpdate.bind(this);
    }

    public insertUpdate() {
        const orderid: HTMLInputElement = document.getElementById('OrderID')
as HTMLInputElement;
        const cusid: HTMLInputElement = document.getElementById('CustomerID')
as HTMLInputElement;
        const empid: HTMLInputElement = document.getElementById('EmployeeID')
as HTMLInputElement;
        const rowdata: { OrderID: number, CustomerID: string, EmployeeID:
number } = {
            CustomerID: cusid.value,
            EmployeeID: +empid.value,
            OrderID: +orderid.value
        };
        if (!rowdata.OrderID) { return; }
        this.dm.update('OrderID', rowdata);
        this.dm.executeQuery(new Query())
        .then((e: ReturnOption) => {
            this.setState({
                items: (e.result as object[]).map((row:
object, index: number) => (
                    <Row key={index} {...row} />
                ))
            });
        });
    }

    public render() {
        return <div><div style={this.style}>
            <input type="number" id='OrderID' placeholder="Order ID" />
            <input type="text" id="CustomerID" placeholder="Customer ID" />
            <input type="number" id="EmployeeID" placeholder="Employee ID" />
            <input type="button" value="Update" id="manipulate"
onClick={this.insertUpdate} /></div>
            <div><table id='datatable' className='e-table'>
                <thead>
                    <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
                </thead>
                <tbody>{ getValue('items', this.state) }</tbody>
            </table>
            </div></div>
        }
    }
}

```

ROWTEMPLATE.JSX

```
import * as React from 'react';
export class Row extends React.Component {
  render() {
    const item = this.props;
    return (
      <tr>
        <td>{item.OrderID}</td>
        <td>{item.CustomerID}</td>
        <td>{item.EmployeeID}</td>
      </tr>
    );
  }
}
```

ROWTEMPLATE.TSX

```
import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
  public render() {
    const item: IOrders = this.props as IOrders;
    return (
      <tr>
        <td>{item.OrderID}</td>
        <td>{item.CustomerID}</td>
        <td>{item.EmployeeID}</td>
      </tr>
    );
  }
}
```

Primary key name is required by the **update** method to find the record to be updated.

Remove

The **remove** method of **DataManager** is used to remove a record from the data source. For remote data source, the record details such as primary key and data will be send along with the request to the server.

APP.JSX

```
import { getValue } from '@syncfusion/ej2-base';
import { DataManager, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { Row } from './rowTemplate';
export default class App extends React.Component {
  dm;
  style;
  constructor(props) {
    super(props);
    this.state = { items: [] };
    this.style = { class: 'e-form' };
    this.dm = new DataManager(data.slice(0, 5));
    this.dm.executeQuery(new Query())
      .then((e) => {
        this.setState({
          items: e.result.map((row, index) => (
            <Row key={index} {...row} />
          ))
        });
      });
  }
}
```

```

    });
    this.insertUpdate = this.insertUpdate.bind(this);
  }
  insertUpdate() {
    const orderid = document.getElementById('OrderID');
    const rowdata = {
      OrderID: +orderid.value,
    };
    if (!rowdata.OrderID) {
      return;
    }
    this.dm.remove('OrderID', rowdata);
    this.dm.executeQuery(new Query())
      .then((e) => {
        this.setState({
          items: e.result.map((row, index) => (
            <Row key={index} {...row} />
          ))
        });
      });
  }
  render() {
    return (<div><div style={this.style}>
      <input type="number" id='OrderID' placeholder="Order ID"/>
      <input type="button" value="Remove" id="manipulate"
onClick={this.insertUpdate}/></div>
      <table id='datatable' className='e-table'>
        <thead>
          <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
        </thead>
        <tbody>{getValue('items', this.state)}</tbody>
      </table>
    </div>);
  }
}

```

APP.TSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, Query, ReturnOption } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { Row } from './rowTemplate';
export default class App extends React.Component<{}, {}>{
  public dm: DataManager;
  public style: { [x: string]: string };
  constructor(props: object) {
    super(props);
    this.state = { items: [] };
    this.style = { class: 'e-form' };
    this.dm = new DataManager(data.slice(0, 5));
    this.dm.executeQuery(new Query())
      .then((e: ReturnOption) => {
        this.setState({
          items: (e.result as object[]).map((row: object, index: number) => (

```

```

        <Row key={index} {...row} />
      ))
    });
  });
  this.insertUpdate = this.insertUpdate.bind(this)
}
public insertUpdate() {
  const orderid: HTMLInputElement = document.getElementById('OrderID')
as HTMLInputElement;
  const rowdata: { OrderID: number } = {
    OrderID: +orderid.value,
  };
  if(!rowdata.OrderID) { return; }
  this.dm.remove('OrderID', rowdata);
  this.dm.executeQuery(new Query())
    .then((e: ReturnOption) => {
      this.setState({
        items: (e.result as object[]).map((row:
object, index: number) => (
          <Row key={index} {...row} />
        ))
      });
    });
}
public render() {
  return (<div><div style={this.style}>
    <input type="number" id='OrderID' placeholder="Order ID"/>
    <input type="button" value="Remove" id="manipulate"
onClick={this.insertUpdate} /></div>
    <table id='datatable' className='e-table'>
      <thead>
        <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
      </thead>
      <tbody>{ getValue('items', this.state) }</tbody>
    </table>
  </div>)
}
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
export class Row extends React.Component {
  render() {
    const item = this.props;
    return (<tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>);
  }
}

```

ROWTEMPLATE.TSX

```
import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
  public render() {
    const item: IOrders = this.props as IOrders;
    return <tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>
  }
}
```

Primary key name and its value are required to find the record to be removed.

Batch Edit Operation

[DataManager](#) supports batch processing for the CRUD operations. You can use the [saveChanges](#) method to batch the edit operation. For remote data source, requests to add, remove and change are handled altogether at a time rather than passing the request separately for each operation.

APP.JSX

```
import { getValue } from '@syncfusion/ej2-base';
import { DataManager, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { Row } from './rowTemplate';
export default class App extends React.Component {
  dm;
  style;
  changes;
  constructor(props) {
    super(props);
    this.state = { items: [] };
    this.style = { class: 'e-form' };
    this.changes = { changedRecords: [], addedRecords: [],
    deletedRecords: [] };
    this.dm = new DataManager(data.slice(0, 5));
    this.dm.executeQuery(new Query())
    .then((e) => {
      this.setState({
        items: e.result.map((row) => (
          <Row key={row.OrderID} {...row} />
        ))
      });
    });
    this.action = this.action.bind(this);
    this.saveChanges = this.saveChanges.bind(this);
  }
  action(event) {
    let action = event.currentTarget.getAttribute('value');
    action = (action === 'Update' ? 'changed' : action === 'Insert' ?
    'added' : 'deleted') + 'Records';
    const orderid = document.getElementById('OrderID');
    const cusid = document.getElementById('CustomerID');
    const empid = document.getElementById('EmployeeID');
```

```

        const rowdata = {
            CustomerID: cusid.value,
            EmployeeID: +empid.value,
            OrderID: +orderid.value
        };
        if (!rowdata.OrderID) {
            return;
        }
        this.changes[action].push(rowdata);
        orderid.value = cusid.value = empid.value = '';
    }
    saveChanges() {
        this.dm.saveChanges(this.changes);
        this.dm.executeQuery(new Query())
            .then((e) => {
                this.setState({
                    items: e.result.map((row) => (
                        <Row key={row.OrderID} {...row} />
                    ))
                });
            });
        this.changes = { changedRecords: [], addedRecords: [],
            deletedRecords: [] };
    }
    render() {
        return (<div><div style={this.style}>
            <input type="number" id='OrderID' placeholder="Order ID"/>
            <input type="text" id="CustomerID" placeholder="Customer ID"/>
            <input type="number" id="EmployeeID" placeholder="Employee ID"/>
            <input type="button" value="Insert" onClick={this.action}/>
            <input type="button" value="Update" onClick={this.action}/>
            <input type="button" value="Remove" onClick={this.action}/></div>
            <div style={this.style}>
                <label>Click to Save changes:</label>
                <input type="button" value="Save Changes"
onClick={this.saveChanges}/></div>
            <div><table id='datatable' className='e-table'>
                <thead>
                    <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
                </thead>
                <tbody>{getValue('items', this.state)}</tbody>
            </table></div></div>);
    }
}

```

APP.TSX

```

import { getValue } from '@syncfusion/ej2-base';
import { DataManager, Query, ReturnOption } from '@syncfusion/ej2-data';
import * as React from 'react';
import { data } from './datasource';
import { Row } from './rowTemplate';
export default class App extends React.Component<{}, {}>{
    public dm: DataManager;
    public style: { [x: string]: string };
}

```



```

public changes: {
    addedRecords: object[],
    changedRecords: object[],
    deletedRecords: object[]
};
constructor(props: object) {
    super(props);
    this.state = { items: [] };
    this.style = { class: 'e-form' };
    this.changes = { changedRecords: [], addedRecords: [],
deletedRecords: [] };
    this.dm = new DataManager(data.slice(0, 5));
    this.dm.executeQuery(new Query())
    .then((e: ReturnOption) => {
        this.setState({
            items: (e.result as object[]).map((row: object) => (
                <Row key={row.OrderID} {...row} />
            ))
        });
    });
    // this.dm.executeQuery(new Query())
    //     .then((e: ReturnOption) => {
    //         this.setState({
    //             items: (e.result as object[]).map((row: object) =>
    (<Row {...row}/>))
    //         });
    //     });
    this.action = this.action.bind(this);
    this.saveChanges = this.saveChanges.bind(this);
}
public action(event: React.MouseEvent) {
    let action: string = event.currentTarget.getAttribute('value') as
string;
    action = (action === 'Update' ? 'changed' : action === 'Insert' ?
'added' : 'deleted') + 'Records';
    const orderid: HTMLInputElement = document.getElementById('OrderID')
as HTMLInputElement;
    const cusid: HTMLInputElement = document.getElementById('CustomerID')
as HTMLInputElement;
    const empid: HTMLInputElement = document.getElementById('EmployeeID')
as HTMLInputElement;
    const rowdata: { OrderID: number, CustomerID: string, EmployeeID:
number } = {
        CustomerID: cusid.value,
        EmployeeID: +empid.value,
        OrderID: +orderid.value
    };
    if (!rowdata.OrderID) { return; }
    this.changes[action].push(rowdata);
    orderid.value = cusid.value = empid.value = '';
}
public saveChanges(): void {
    this.dm.saveChanges(this.changes);
    this.dm.executeQuery(new Query())
        .then((e: ReturnOption) => {
            this.setState({
                items: (e.result as object[]).map((row: object) => (

```

```

                <Row key={row.OrderID} {...row} />
            ))
        });
    });
    this.changes = { changedRecords: [], addedRecords: [],
deletedRecords: [] };
}
    public render() {
        return (<div><div style={this.style}>
            <input type="number" id='OrderID' placeholder="Order ID" />
            <input type="text" id="CustomerID" placeholder="Customer ID" />
            <input type="number" id="EmployeeID" placeholder="Employee ID" />
            <input type="button" value="Insert" onClick={this.action} />
            <input type="button" value="Update" onClick={this.action} />
            <input type="button" value="Remove" onClick={this.action}
/></div>
            <div style={this.style}>
                <label>Click to Save changes:</label>
                <input type="button" value="Save Changes"
onClick={this.saveChanges} /></div>
            <div><table id='datatable' className='e-table'>
                <thead>
                    <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
                </thead>
                <tbody>{getValue('items', this.state)}</tbody>
            </table></div></div>
        )
    }
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
export class Row extends React.Component {
    render() {
        const item = this.props;
        return (<tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>);
    }
}

```

ROWTEMPLATE.TSX

```

import * as React from 'react';
import { IOOrders } from './orders';
export class Row extends React.Component<{}, {}>{
    public render() {
        const item: IOOrders = this.props as IOOrders;
        return (<tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>);
    }
}

```

```
}
}
```

How to in React Data component

Work in offline mode

On remote data binding, every time invoking **executeQuery** will send request to the server and the query will be processed on server-side. To avoid post back to server on calling **executeQuery** to load all the data on initialization time and make the query processing in client-side. To enable this behavior, you can use **offline** property of [DataManager](#).

APP.JSX

```
import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { Row } from './rowTemplate';
const SERVICE_URI =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = { items: [] };
  }
  componentDidMount() {
    const dm = new DataManager({ url: SERVICE_URI, adaptor: new
ODataV4Adaptor(), offline: true }, new Query().take(8));
    dm.ready.then((e) => {
      const res = e.result.map((row) => <Row key={row.OrderID} {...row}
/>);
      this.setState({
        items: res,
      });
    });
  }
  render() {
    return (<table id='datatable' className='e-table'>
      <thead>
        <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
      </thead>
      <tbody>{getValue('items', this.state)}</tbody>
    </table>);
  }
}
```

APP.TSX

```
import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query, ReturnOption } from
 '@syncfusion/ej2-data';
import * as React from 'react';
import { IOrders } from './orders';
import { Row } from './rowTemplate';
```

```

const SERVICE_URI: string =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component<{}, {}>{
  constructor(props: object) {
    super(props);
    this.state = { items: [] };
  }
  componentDidMount() {
    const dm: DataManager = new DataManager({ url: SERVICE_URI, adaptor:
new ODataV4Adaptor(), offline: true }, new Query().take(8));
    dm.ready.then((e: ReturnOption) => {
      const res = (e.result as IOOrders[]).map((row: IOOrders, index:
number) => (
        <Row key={row.OrderID} {...row} />
      ));
      this.setState({
        items: res
      });
    });
  }
  public render() {
    return <table id='datatable' className='e-table'>
      <thead>
        <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
      </thead>
      <tbody>{ getValue('items', this.state) }</tbody>
    </table>
  }
}

```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```

export interface IOOrders {
  OrderID: number;
  EmployeeID: number;
  CustomerID: string;
  Order_Details: object[];
}

```

ROWTEMPLATE.JSX

```

import * as React from 'react';
export class Row extends React.Component {
  render() {
    const item = this.props;
    return <tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>;
  }
}

```

```
}
}
```

ROWTEMPLATE.TSX

```
import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
    public render() {
        const item: IOrders = this.props as IOrders;
        return (<tr>
            <td>{item.OrderID}</td>
            <td>{item.CustomerID}</td>
            <td>{item.EmployeeID}</td>
        </tr>)
    }
}
```

The loaded data will be cached in the **json** property of [DataManager](#).

Sending additional parameters to server

You can use the [addParams](#) method of [Query](#) class, to add custom parameter to the data request.

APP.JSX

```
import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query } from '@syncfusion/ej2-data';
import * as React from 'react';
import { Row } from './rowTemplate';
const SERVICE_URI =
'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component {
    constructor(props) {
        super(props);
        this.state = { items: [] };
    }
    componentDidMount() {
        new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
            .executeQuery(new Query().addParams('$top', '8'))
            .then((e) => {
                const res = e.result.map((row) => <Row key={row.OrderID} {...row}
/>);
                this.setState({
                    items: res,
                });
            });
    }
    render() {
        return (<table id='datatable' className='e-table'>
            <thead>
                <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
            </thead>
            <tbody>{getValue('items', this.state)}</tbody>
        </table>);
    }
}
```

```
}

```

APP.TSX

```
import { getValue } from '@syncfusion/ej2-base';
import { DataManager, ODataV4Adaptor, Query, ReturnOption } from
 '@syncfusion/ej2-data';
import * as React from 'react';
import { IOrders } from './orders';
import { Row } from './rowTemplate';
const SERVICE_URI: string =
 'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
export default class App extends React.Component<{}, {}>{
  constructor(props: object) {
    super(props);
    this.state = { items: [] };
  }
  componentDidMount() {
    new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor() })
      .executeQuery(new Query().addParams('$top', '8'))
      .then((e: ReturnOption) => {
        const res = (e.result as IOrders[]).map((row: IOrders, index:
number) => (
          <Row key={row.OrderID} {...row} />
        ));
        this.setState({
          items: res
        });
      });
  }
  public render() {
    return (<table id='datatable' className='e-table'>
      <thead>
        <tr><th>Order ID</th><th>Customer ID</th><th>Employee
ID</th></tr>
      </thead>
      <tbody>{ getValue('items', this.state) }</tbody>
    </table>)
  }
}
```

ORDERS.JSX

```
export {};
```

ORDERS.TSX

```
export interface IOrders {
  OrderID: number;
  EmployeeID: number;
  CustomerID: string;
  Order_Details: object[];
}
```

ROWTEMPLATE.JSX

```
import * as React from 'react';
export class Row extends React.Component {
  render() {
    const item = this.props;
    return (<tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>);
  }
}
```

ROWTEMPLATE.TSX

```
import * as React from 'react';
import { IOrders } from './orders';
export class Row extends React.Component<{}, {}>{
  public render() {
    const item: IOrders = this.props as IOrders;
    return (<tr>
      <td>{item.OrderID}</td>
      <td>{item.CustomerID}</td>
      <td>{item.EmployeeID}</td>
    </tr>)
  }
}
```

Adding custom headers

You can add custom headers to the request made by [DataManager](#) using the **headers** property.

```
`ts
```

```
import { DataManager, ODataV4Adaptor, Query, ReturnOption } from '@syncfusion/ej2-data';
const SERVICE_URI: string = 'https://services.odata.org/V4/Northwind/Northwind.svc/Orders/';
new DataManager({ url: SERVICE_URI, adaptor: new ODataV4Adaptor, headers:[{ 'syncfusion': 'true' } ] })
.executeQuery(new Query())
.then((e: ReturnOption) => {
// get result from e.result
});
`
```

Adding custom headers while making cross domain request will initiate preflight request.

DatePicker

Getting started

This section explains you the steps required to create a simple [React DatePickerLink to the Video](#) and demonstrate the basic usage of the DatePicker component.

To get start quickly with React Date Picker, you can check on this video:

Dependencies

The below list of dependencies are required to use the `DatePicker` component in your application.

```
`javascript
|-- @syncfusion/ej2-react-calendars
|-- @syncfusion/ej2-react-base
|-- @syncfusion/ej2-base
|-- @syncfusion/ej2-data
|-- @syncfusion/ej2-calendars
|-- @syncfusion/ej2-inputs
|-- @syncfusion/ej2-splitbuttons
|-- @syncfusion/ej2-lists
|-- @syncfusion/ej2-popups
|-- @syncfusion/ej2-buttons
`,`
```

Installation and configuration

You can use [create-react-app](#) to setup the applications.

To install `create-react-app` run the following command.

```
`
npm install -g create-react-app
`,`
```

- To setup basic `React` sample use following commands.

```
`
create-react-app quickstart --scripts-version=react-scripts-ts
cd quickstart
`,`
```

Adding Syncfusion packages

All the available Essential JS 2 packages are published in [npmjs.com](#) public registry.

You can choose the component that you want to install. For this application, we are going to use `DatePicker` component.

To install `DatePicker` component, use the following command

```
`bash
npm install @syncfusion/ej2-react-calendars --save
```


,

Adding Style sheet to the Application

To render the DatePicker component, need to import DatePicker and its dependent component's styles as given below in `App.css`.

```
`css
@import "../node_modules/@syncfusion/ej2-base/styles/material.css";
@import "../node_modules/@syncfusion/ej2-buttons/styles/material.css";
@import "../node_modules/@syncfusion/ej2-inputs/styles/material.css";
@import "../node_modules/@syncfusion/ej2-popups/styles/material.css";
@import "../node_modules/@syncfusion/ej2-react-calendars/styles/material.css";
,
```

Note: If you want to refer the combined component styles, please make use of our [CRG](#) (Custom Resource Generator) in your application.

Adding DatePicker component to the Application

- To include the DatePicker component in application import the `DatePickerComponent` from `ej2-react-calendars` package in `App.tsx`.
- Then add the DatePicker component as shown in below code example.

[src/App.tsx]

[Class-component]

```
`ts
// import the datePickerComponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import './App.css';
export default class App extends React.Component<{}, {}> {
  public render() {
    return <DatePickerComponent id="datepicker" />;
  }
}
```

,

[Functional-component]

```
`ts
// import the datePickerComponent
```

```
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  return <DatePickerComponent id="datepicker" />;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

Run the application

Now run the `npm start` command in the console, it will run your application and open the browser window.

```
npm start
```

The below examples shows the basic DatePicker component.

[Class-component]

INDEX.JSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  render() {
    return <DatePickerComponent id="datepicker" placeholder="Enter
date"/>;
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
  public render() {
    return <DatePickerComponent id="datepicker" placeholder="Enter
date"/>;
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]

INDEX.JSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    return <DatePickerComponent id="datepicker" placeholder="Enter date"/>;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App(){
    return <DatePickerComponent id="datepicker" placeholder="Enter date"/>;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

Setting the value, min and max dates

The following example demonstrates how to set the value, min and max dates on initializing the DatePicker.

Here the DatePicker allows to select a date within a range from 9th to 15th in a month of May 2017. To know more about range restriction in DatePicker, please refer this [page](#).

[Class-component]**INDEX.JSX**

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    // initialize the value, min and max
    dateValue = new Date('05/11/2017');
    minDate = new Date('05/09/2017');
    maxDate = new Date('05/15/2017');
    render() {
        return <DatePickerComponent id="datepicker" value={this.dateValue}
min={this.minDate} max={this.maxDate}/>;
    }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
```

```
export default class App extends React.Component<{}, {}> {
  // initialize the value, min and max
  private dateValue: Date = new Date('05/11/2017');
  private minDate: Date = new Date('05/09/2017');
  private maxDate: Date = new Date('05/15/2017');
  public render() {
    return <DatePickerComponent id="datepicker" value={this.dateValue}
    min={this.minDate} max={this.maxDate} />;
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]

INDEX.JSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  // initialize the value, min and max
  const dateValue = new Date('05/11/2017');
  const minDate = new Date('05/09/2017');
  const maxDate = new Date('05/15/2017');
  return <DatePickerComponent id="datepicker" value={dateValue}
  min={minDate} max={maxDate}/>;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  // initialize the value, min and max
  const dateValue: Date = new Date('05/11/2017');
  const minDate: Date = new Date('05/09/2017');
  const maxDate: Date = new Date('05/15/2017');
  return <DatePickerComponent id="datepicker" value={dateValue}
  min={minDate} max={maxDate} />;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

See Also

- [Change the format of selected date](#)
- [Render DatePicker with specific culture](#)
- [How to change the initial view of the DatePicker](#)
- [How to achieve dynamic form validation with DatePicker](#)

Note: You can also explore our [React DatePicker example](#) that shows you how to render the DatePicker in React.

Date range in React DatePicker component

DatePicker provides an option to select a date value within a specified range by using the [min](#) and [max](#) properties. Always the min value has to be lesser than the max value.

When the min and max properties are configured and the selected date value is out-of-range or invalid, then the model value will be set to **out of range** date value or **null** respectively with highlighted **error** class to indicates the date is out of range or invalid.

The value property depends on the min/max with respect to [strictMode](#) property. The below example allows to select a date within a range from 7th to 27th days in a month.

[Class-component]

INDEX.JSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    // creates a datepicker with min and max property
    minDate = new Date(new Date().getFullYear(), new Date().getMonth(), 7);
    maxDate = new Date(new Date().getFullYear(), new Date().getMonth(), 27);
    dateValue = new Date(new Date().setDate(14));
    render() {
        return <DatePickerComponent id="datepicker" value={this.dateValue}
min={this.minDate} max={this.maxDate}/>;
    }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    // creates a datepicker with min and max property
    private minDate: Date = new Date(new Date().getFullYear(), new
Date().getMonth(), 7);
    private maxDate: Date = new Date(new Date().getFullYear(), new
Date().getMonth(), 27);
    private dateValue: Date = new Date(new Date().setDate(14));
    public render() {
        return <DatePickerComponent id="datepicker" value={this.dateValue}
min={this.minDate} max={this.maxDate} />;
    }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]

INDEX.JSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // creates a datepicker with min and max property
    const minDate = new Date(new Date().getFullYear(), new Date().getMonth(),
7);
    const maxDate = new Date(new Date().getFullYear(), new Date().getMonth(),
27);
    const dateValue = new Date(new Date().setDate(14));
    return <DatePickerComponent id="datepicker" value={dateValue}
min={minDate} max={maxDate}/>;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    // creates a datepicker with min and max property
    const minDate: Date = new Date(new Date().getFullYear(), new
Date().getMonth(), 7);
    const maxDate: Date = new Date(new Date().getFullYear(), new
Date().getMonth(), 27);
    const dateValue: Date = new Date(new Date().setDate(14));
    return <DatePickerComponent id="datepicker" value={dateValue}
min={minDate} max={maxDate} />;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

If the value of `min` or `max` properties changed through code behind. Then you have to update the `value` property to set within the range.

Date format in React DatePicker component

Date format is a way of representing the date value in different string format in the textbox.

By default, the DatePicker's format is based on the culture. You can also set the own custom format by using the

[format](#) property.

Once the date format property has been defined it will be common to all the cultures.

To know more about the date format standards, refer to the [Internationalization Date Format](#) section.

The following example demonstrates the DatePicker with the custom format (yyyy-MM-dd).

[Class-component]

INDEX.JSX

```
// import the datePickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    dateValue = new Date();
    render() {
        return <DatePickerComponent id="datepicker" value={this.dateValue}
format='yyyy-MM-dd' placeholder='Enter date' />;
    }
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datePickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    private dateValue:Date=new Date();
    public render() {
        return <DatePickerComponent id="datepicker" value={this.dateValue}
format='yyyy-MM-dd' placeholder='Enter date' />
    }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]**INDEX.JSX**

```
// import the datePickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const dateValue = new Date();
    return <DatePickerComponent id="datepicker" value={dateValue}
format='yyyy-MM-dd' placeholder='Enter date' />;
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datePickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    const dateValue:Date=new Date();
    return <DatePickerComponent id="datepicker" value={dateValue}
format='yyyy-MM-dd' placeholder='Enter date' />
}
```

```
};  
ReactDOM.render(<App />, document.getElementById('element'));
```

Date masking in React Datepicker component

DatePicker has `enableMask` property that provides the option to enable the built-in date masking support. Also, you must inject the MaskedDateTime module to enable the masking support.

[Class-component]

INDEX.JSX

```
import { DatePickerComponent, Inject, MaskedDateTime } from '@syncfusion/ej2-react-calendars';  
import * as ReactDOM from 'react-dom';  
import * as React from 'react';  
export default class App extends React.Component {  
  render() {  
    return <DatePickerComponent enableMask={true}><Inject  
services={[MaskedDateTime]} /></DatePickerComponent>;  
  }  
}  
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import { DatePickerComponent, Inject, MaskedDateTime } from '@syncfusion/ej2-react-calendars';  
import * as ReactDOM from 'react-dom';  
import * as React from 'react';  
export default class App extends React.Component<{}, {}> {  
  public render() {  
    return <DatePickerComponent enableMask={true}><Inject  
services={[MaskedDateTime]} /></DatePickerComponent>  
  }  
}  
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]

INDEX.JSX

```
import { DatePickerComponent, Inject, MaskedDateTime } from '@syncfusion/ej2-react-calendars';  
import * as ReactDOM from 'react-dom';  
import * as React from 'react';  
function App() {  
  return <DatePickerComponent enableMask={true}><Inject  
services={[MaskedDateTime]} /></DatePickerComponent>;  
}  
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX


```
import { DatePickerComponent, Inject, MaskedDateTime } from '@syncfusion/ej2-react-calendars';
import * as ReactDOM from 'react-dom';
import * as React from 'react';
function App() {
    return <DatePickerComponent enableMask={true}><Inject
services={[MaskedDateTime]} /></DatePickerComponent>
}
ReactDOM.render(<App />, document.getElementById('element'));
```

The mask pattern is defined based on the provided date format to the component. If the format is not specified, the mask pattern is formed based on the default format of the current culture.

| **Keys | Actions |**

| --- | --- |

| Up / Down arrows | To increment and decrement the selected portion of the date. |

| Left / Right arrows and Tab | To navigate the selection from one portion to next portion |

The following example demonstrates default and custom format of DatePicker component with mask.

[Class-component]

INDEX.JSX

```
import { DatePickerComponent, Inject, MaskedDateTime } from '@syncfusion/ej2-react-calendars';
import * as ReactDOM from 'react-dom';
import * as React from 'react';
export default class App extends React.Component {
    render() {
        return (<div>
            /* specifies the masked DatePicker component without format */
            <DatePickerComponent enableMask={true}><Inject
services={[MaskedDateTime]} /></DatePickerComponent>
            <br />
            <br />
            /* specifies the masked DatePicker component with format */
            <DatePickerComponent format='dd-MM-yyyy'
enableMask={true}><Inject services={[MaskedDateTime]} /></DatePickerComponent>
            </div>);
    }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import { DatePickerComponent, Inject, MaskedDateTime } from '@syncfusion/ej2-react-calendars';
import * as ReactDOM from 'react-dom';
import * as React from 'react';
export default class App extends React.Component<{}, {}> {
    public render() {
```

```

        return (
            <div>
                /* specifies the masked DatePicker component without format */
                <DatePickerComponent enableMask={true}><Inject
services={ [MaskedDateTime] } /></DatePickerComponent>
                <br />
                <br />
                /* specifies the masked DatePicker component with format */
                <DatePickerComponent format='dd-MM-yyyy'
enableMask={true}><Inject services={ [MaskedDateTime] }
/></DatePickerComponent>
            </div>
        );
    }
}
ReactDOM.render(<App />, document.getElementById('element'));

```

[Functional-component]

INDEX.JSX

```

import { DatePickerComponent, Inject, MaskedDateTime } from '@syncfusion/ej2-react-calendars';
import * as ReactDOM from 'react-dom';
import * as React from 'react';
function App() {
    return (<div>
        /* specifies the masked DatePicker component without format */
        <DatePickerComponent enableMask={true}><Inject
services={ [MaskedDateTime] } /></DatePickerComponent>
        <br />
        <br />
        /* specifies the masked DatePicker component with format */
        <DatePickerComponent format='dd-MM-yyyy'
enableMask={true}><Inject services={ [MaskedDateTime] } /></DatePickerComponent>
    </div>);
}
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { DatePickerComponent, Inject, MaskedDateTime } from '@syncfusion/ej2-react-calendars';
import * as ReactDOM from 'react-dom';
import * as React from 'react';
function App() {
    return (
        <div>
            /* specifies the masked DatePicker component without format */
            <DatePickerComponent enableMask={true}><Inject
services={ [MaskedDateTime] } /></DatePickerComponent>

```

```

        <br />
        <br />
        { /* specifies the masked DatePicker component with format
    */}
        <DatePickerComponent format='dd-MM-yyyy'
enableMask={true}><Inject services={ [MaskedDateTime] }
/></DatePickerComponent>
    </div>
    );
}
ReactDOM.render(<App />, document.getElementById('element'));

```

Configure Mask Placeholder

You can change mask placeholder value through property `maskPlaceholder`. By default , it takes the full name of date and time co-ordinates such as `day`, `month`, `year`, `hour` etc.

While changing to a culture other than `English`, ensure that locale text for the concerned culture is loaded through load method of L10n class for mask placeholder values like below.

```

`ts
//Load the L10n from ej2-base
import { L10n } from '@syncfusion/ej2-base';
//load the locale object to set the localized mask placeholder value
L10n.load({
  'de': {
    datepicker: { day: 'Tag', month: 'Monat', year: 'Jahr' }
  }
});
`

```

The following example demonstrates default and customized mask placeholder value.

[Class-component]

INDEX.JSX

```

import { DatePickerComponent, Inject, MaskedDateTime } from '@syncfusion/ej2-react-calendars';
import * as ReactDOM from 'react-dom';
import * as React from 'react';
export default class App extends React.Component {
  maskPlaceholderValue;
  constructor(props) {
    super(props);
    this.maskPlaceholderValue = { day: 'd', month: 'M', year: 'y' };
  }
  render() {
    return (
      <div>
        { /* specifies the masked DatePicker component without mask
placeholder */}

```

```

        <DatePickerComponent enableMask={true}><Inject
services={ [MaskedDateTime] }/></DatePickerComponent>
        <br />
        <br />
        /* specifies the masked DatePicker component with mask
placeholder */
        <DatePickerComponent enableMask={true}
maskPlaceholder={this.maskPlaceholderValue}><Inject
services={ [MaskedDateTime] }/></DatePickerComponent>
        </div>;
    }
}
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { DatePickerComponent, Inject, MaskedDateTime } from '@syncfusion/ej2-
react-calendars';
import * as ReactDOM from 'react-dom';
import * as React from 'react';
export default class App extends React.Component<{}, {}> {
    private maskPlaceholderValue: object;
    constructor(props: {}) {
        super(props);
        this.maskPlaceholderValue = {day: 'd', month: 'M', year: 'y'};
    }
    public render() {
        return (
            <div>
                /* specifies the masked DatePicker component without mask
placeholder */
                <DatePickerComponent enableMask={true}><Inject
services={ [MaskedDateTime] } /></DatePickerComponent>
                <br />
                <br />
                /* specifies the masked DatePicker component with mask
placeholder */
                <DatePickerComponent enableMask={true}
maskPlaceholder={this.maskPlaceholderValue}><Inject
services={ [MaskedDateTime] } /></DatePickerComponent>
                </div>
            );
        );
    }
}
ReactDOM.render(<App />, document.getElementById('element'));

```

[Functional-component]**INDEX.JSX**

```

import { DatePickerComponent, Inject, MaskedDateTime } from '@syncfusion/ej2-
react-calendars';
import * as ReactDOM from 'react-dom';
import * as React from 'react';
function App() {
    let maskPlaceholderValue;

```

```

    maskPlaceholderValue = { day: 'd', month: 'M', year: 'y' };
    return (<div>
        /* specifies the masked DatePicker component without mask
placeholder */
        <DatePickerComponent enableMask={true}><Inject
services={ [MaskedDateTime] }/></DatePickerComponent>
        <br />
        <br />
        /* specifies the masked DatePicker component with mask
placeholder */
        <DatePickerComponent enableMask={true}
maskPlaceholder={maskPlaceholderValue}><Inject
services={ [MaskedDateTime] }/></DatePickerComponent>
        </div>);
}
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { DatePickerComponent, Inject, MaskedDateTime } from '@syncfusion/ej2-
react-calendars';
import * as ReactDOM from 'react-dom';
import * as React from 'react';
function App() {
    let maskPlaceholderValue: object;
    maskPlaceholderValue = {day: 'd', month: 'M', year: 'y'};
    return (
        <div>
            /* specifies the masked DatePicker component without mask
placeholder */
            <DatePickerComponent enableMask={true}><Inject
services={ [MaskedDateTime] } /></DatePickerComponent>
            <br />
            <br />
            /* specifies the masked DatePicker component with mask
placeholder */
            <DatePickerComponent enableMask={true}
maskPlaceholder={maskPlaceholderValue}><Inject services={ [MaskedDateTime] }
/></DatePickerComponent>
            </div>
        );
    }
ReactDOM.render(<App />, document.getElementById('element'));

```

Globalization in React DatePicker component

Globalization is the combination of internalization and localization. You can adapt the component to various languages by parsing and formatting the date or number [Internationalization](#) and also add culture specific customization and translation to the text [localization](#).

By default, DatePicker date format, week, and month names are specific to the **American English** culture. It utilizes the [Essential JavaScript 2 Internationalization](#) package to parse and format the date object based on the culture by using the official [UNICODE CLDR](#) JSON data. It provides the `loadCldr` method to load culture specific CLDR JSON data. To use a different culture other than **English**, follow the steps below:

- Install the `CLDR-Data` package by using the following command (installs all the CLDR JSON data). To know more about CLDR-Data refer to the [CLDR-Data](#) link.

`

```
npm install cldr-data --save
```

`

Once the package installed, you can find the culture specific JSON data under the location `\node_modules\cldr-data`.

- Now import the installed CLDR JSON data into the `app.ts` file.
- Now use the [loadCldr](#) method to load the culture specific CLDR JSON data from the installed location to `app.ts` file.
- DatePicker displayed **Sunday** as the first day of week based on default culture ("en-US"). If you want to display the DatePicker with loaded culture's first day of week, you need to import `weekdata.json` file from the `cldr-data/supplemental` as given in the code example.

`ts

```
import { loadCldr } from "@syncfusion/ej2-base";
import * as gregorian from 'cldr-data/main/de/ca-gregorian.json';
import * as numbers from 'cldr-data/main/de/numbers.json';
import * as timeZoneNames from 'cldr-data/main/de/timeZoneNames.json';
import * as numberingSystems from 'cldr-data/supplemental/numberingSystems.json';
import * as weekData from 'cldr-data/supplemental/weekData.json'; // To load the culture based first day of week

loadCldr(numberingSystems, gregorian, numbers, timeZoneNames, weekData);
```

`

if you are facing the error `/node_modules/cldr-data/main/de/*.json (1,1): unused expression, expected an assignment or function call` when you are adding the json files to render the culture sample, then add the below configuration in your `tslint.json` file

`ts

```
"linterOptions": {
  "exclude": [
    "*.json",
    "/*.json"
  ]
}
```

`

The **Localization** library allows you to localize default text content of the DatePicker. The DatePicker component has static text for **today** feature that can be changed to other cultures (Arabic, Deutsch, French, etc.) by defining the [locale](#) value and translation object.

Locale keywords | Text

today | Name of the button to choose Today date.

placeholder | Hint to describe expected value in input element.

- Before changing to a culture other than **English**, ensure that locale text for the concerned culture is loaded through **load** method of **L10n** class.

```
`ts
//Load the L10n from ej2-base
import { L10n } from "@syncfusion/ej2-base";
//load the locale object to set the localized placeholder value
L10n.load({
  de: {
    datepicker: {
      placeholder: "Wählen Sie ein Datum",
      today:"heute"
    }
  }
});
`
```

- Set the culture by using the [locale](#) property. The below code example, initialize the DatePicker component in **German** culture with corresponding localized text.

The following example demonstrates the DatePicker in **German** culture.

[Class-component]

CA-GREGORIAN.JSON

```
{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 12879 $",
          "_cldrVersion": "30.0.3"
        },
        "language": "de"
      },
      "dates": {
```

```
"calendars": {
  "gregorian": {
    "months": {
      "format": {
        "abbreviated": {
          "1": "Jan.",
          "2": "Feb.",
          "3": "März",
          "4": "Apr.",
          "5": "Mai",
          "6": "Juni",
          "7": "Juli",
          "8": "Aug.",
          "9": "Sep.",
          "10": "Okt.",
          "11": "Nov.",
          "12": "Dez."
        },
        "narrow": {
          "1": "J",
          "2": "F",
          "3": "M",
          "4": "A",
          "5": "M",
          "6": "J",
          "7": "J",
          "8": "A",
          "9": "S",
          "10": "O",
          "11": "N",
          "12": "D"
        },
        "wide": {
          "1": "Januar",
          "2": "Februar",
          "3": "März",
          "4": "April",
          "5": "Mai",
          "6": "Juni",
          "7": "Juli",
          "8": "August",
          "9": "September",
          "10": "Oktober",
          "11": "November",
          "12": "Dezember"
        }
      },
      "stand-alone": {
        "abbreviated": {
          "1": "Jan",
          "2": "Feb",
          "3": "Mär",
          "4": "Apr",
          "5": "Mai",
          "6": "Jun",
          "7": "Jul",
          "8": "Aug",
```



```

        "9": "Sep",
        "10": "Okt",
        "11": "Nov",
        "12": "Dez"
    },
    "narrow": {
        "1": "J",
        "2": "F",
        "3": "M",
        "4": "A",
        "5": "M",
        "6": "J",
        "7": "J",
        "8": "A",
        "9": "S",
        "10": "O",
        "11": "N",
        "12": "D"
    },
    "wide": {
        "1": "Januar",
        "2": "Februar",
        "3": "März",
        "4": "April",
        "5": "Mai",
        "6": "Juni",
        "7": "Juli",
        "8": "August",
        "9": "September",
        "10": "Oktober",
        "11": "November",
        "12": "Dezember"
    }
    },
    "days": {
        "format": {
            "abbreviated": {
                "sun": "So.",
                "mon": "Mo.",
                "tue": "Di.",
                "wed": "Mi.",
                "thu": "Do.",
                "fri": "Fr.",
                "sat": "Sa."
            },
            "narrow": {
                "sun": "S",
                "mon": "M",
                "tue": "D",
                "wed": "M",
                "thu": "D",
                "fri": "F",
                "sat": "S"
            },
            "short": {
                "sun": "So.",

```

```

        "mon": "Mo.",
        "tue": "Di.",
        "wed": "Mi.",
        "thu": "Do.",
        "fri": "Fr.",
        "sat": "Sa."
    },
    "wide": {
        "sun": "Sonntag",
        "mon": "Montag",
        "tue": "Dienstag",
        "wed": "Mittwoch",
        "thu": "Donnerstag",
        "fri": "Freitag",
        "sat": "Samstag"
    }
},
"stand-alone": {
    "abbreviated": {
        "sun": "So",
        "mon": "Mo",
        "tue": "Di",
        "wed": "Mi",
        "thu": "Do",
        "fri": "Fr",
        "sat": "Sa"
    },
    "narrow": {
        "sun": "S",
        "mon": "M",
        "tue": "D",
        "wed": "M",
        "thu": "D",
        "fri": "F",
        "sat": "S"
    },
    "short": {
        "sun": "So.",
        "mon": "Mo.",
        "tue": "Di.",
        "wed": "Mi.",
        "thu": "Do.",
        "fri": "Fr.",
        "sat": "Sa."
    },
    "wide": {
        "sun": "Sonntag",
        "mon": "Montag",
        "tue": "Dienstag",
        "wed": "Mittwoch",
        "thu": "Donnerstag",
        "fri": "Freitag",
        "sat": "Samstag"
    }
}
},
"quarters": {

```

```
    "format": {
      "abbreviated": {
        "1": "Q1",
        "2": "Q2",
        "3": "Q3",
        "4": "Q4"
      },
      "narrow": {
        "1": "1",
        "2": "2",
        "3": "3",
        "4": "4"
      },
      "wide": {
        "1": "1. Quartal",
        "2": "2. Quartal",
        "3": "3. Quartal",
        "4": "4. Quartal"
      }
    },
    "stand-alone": {
      "abbreviated": {
        "1": "Q1",
        "2": "Q2",
        "3": "Q3",
        "4": "Q4"
      },
      "narrow": {
        "1": "1",
        "2": "2",
        "3": "3",
        "4": "4"
      },
      "wide": {
        "1": "1. Quartal",
        "2": "2. Quartal",
        "3": "3. Quartal",
        "4": "4. Quartal"
      }
    }
  },
  "dayPeriods": {
    "format": {
      "abbreviated": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "morgens",
        "morning2": "vormittags",
        "afternoon1": "mittags",
        "afternoon2": "nachmittags",
        "evening1": "abends",
        "night1": "nachts"
      },
      "narrow": {
        "midnight": "Mitternacht",
        "am": "vm.",
```

```

        "pm": "nm.",
        "morning1": "morgens",
        "morning2": "vormittags",
        "afternoon1": "mittags",
        "afternoon2": "nachmittags",
        "evening1": "abends",
        "night1": "nachts"
    },
    "wide": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "morgens",
        "morning2": "vormittags",
        "afternoon1": "mittags",
        "afternoon2": "nachmittags",
        "evening1": "abends",
        "night1": "nachts"
    }
},
"stand-alone": {
    "abbreviated": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "Morgen",
        "morning2": "Vormittag",
        "afternoon1": "Mittag",
        "afternoon2": "Nachmittag",
        "evening1": "Abend",
        "night1": "Nacht"
    },
    "narrow": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "Morgen",
        "morning2": "Vormittag",
        "afternoon1": "Mittag",
        "afternoon2": "Nachmittag",
        "evening1": "Abend",
        "night1": "Nacht"
    },
    "wide": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "Morgen",
        "morning2": "Vormittag",
        "afternoon1": "Mittag",
        "afternoon2": "Nachmittag",
        "evening1": "Abend",
        "night1": "Nacht"
    }
}
},
"eras": {

```

```

    "eraNames": {
      "0": "v. Chr.",
      "0-alt-variant": "vor unserer Zeitrechnung",
      "1": "n. Chr.",
      "1-alt-variant": "unserer Zeitrechnung"
    },
    "eraAbbr": {
      "0": "v. Chr.",
      "0-alt-variant": "v. u. Z.",
      "1": "n. Chr.",
      "1-alt-variant": "u. Z."
    },
    "eraNarrow": {
      "0": "v. Chr.",
      "0-alt-variant": "v. u. Z.",
      "1": "n. Chr.",
      "1-alt-variant": "u. Z."
    }
  },
  "dateFormats": {
    "full": "EEEE, d. MMMM y",
    "long": "d. MMMM y",
    "medium": "dd.MM.y",
    "short": "dd.MM.yy"
  },
  "timeFormats": {
    "full": "HH:mm:ss zzzz",
    "long": "HH:mm:ss z",
    "medium": "HH:mm:ss",
    "short": "HH:mm"
  },
  "dateTimeFormats": {
    "full": "{1} 'um' {0}",
    "long": "{1} 'um' {0}",
    "medium": "{1}, {0}",
    "short": "{1}, {0}",
    "availableFormats": {
      "d": "d",
      "E": "ccc",
      "Ed": "E, d.",
      "Ehm": "E h:mm a",
      "EHm": "E, HH:mm",
      "Ehms": "E, h:mm:ss a",
      "EHms": "E, HH:mm:ss",
      "Gy": "y G",
      "GyMMM": "MMM y G",
      "GyMMMd": "d. MMM y G",
      "GyMMMED": "E, d. MMM y G",
      "h": "h 'Uhr' a",
      "H": "HH 'Uhr'",
      "hm": "h:mm a",
      "Hm": "HH:mm",
      "hms": "h:mm:ss a",
      "Hms": "HH:mm:ss",
      "hmsv": "h:mm:ss a v",
      "Hmsv": "HH:mm:ss v",
      "hmv": "h:mm a v",

```

```

    "Hmv": "HH:mm v",
    "M": "L",
    "Md": "d.M.",
    "MEd": "E, d.M.",
    "MMd": "d.MM.",
    "MMdd": "dd.MM.",
    "MMM": "LLL",
    "MMMd": "d. MMM",
    "MMMED": "E, d. MMM",
    "MMMMd": "d. MMMM",
    "MMMMEd": "E, d. MMMM",
    "MMMMMW": "'Woche' W 'im' MMM",
    "MMMMW": "'Woche' W 'im' MMM",
    "ms": "mm:ss",
    "y": "y",
    "yM": "M.y",
    "yMd": "d.M.y",
    "yMEd": "E, d.M.y",
    "yMM": "MM.y",
    "yMMdd": "dd.MM.y",
    "yMMM": "MMM y",
    "yMMMd": "d. MMM y",
    "yMMMED": "E, d. MMM y",
    "yMMMM": "MMMM y",
    "yQQQ": "QQQ y",
    "yQQQQ": "QQQQ y",
    "yw": "'Woche' w 'des' 'Jahres' y",
    "yw": "'Woche' w 'des' 'Jahres' y"
  },
  "appendItems": {
    "Day": "{0} ({2}: {1})",
    "Day-Of-Week": "{0} {1}",
    "Era": "{1} {0}",
    "Hour": "{0} ({2}: {1})",
    "Minute": "{0} ({2}: {1})",
    "Month": "{0} ({2}: {1})",
    "Quarter": "{0} ({2}: {1})",
    "Second": "{0} ({2}: {1})",
    "Timezone": "{0} {1}",
    "Week": "{0} ({2}: {1})",
    "Year": "{1} {0}"
  },
  "intervalFormats": {
    "intervalFormatFallback": "{0} - {1}",
    "d": {
      "d": "d.-d."
    },
    "h": {
      "a": "h 'Uhr' a - h 'Uhr' a",
      "h": "h - h 'Uhr' a"
    },
    "H": {
      "H": "HH-HH 'Uhr'"
    },
    "hm": {
      "a": "h:mm a - h:mm a",
      "h": "h:mm-h:mm a",

```

```

        "m": "h:mm-h:mm a"
    },
    "Hm": {
        "H": "HH:mm-HH:mm 'Uhr'",
        "m": "HH:mm-HH:mm 'Uhr'"
    },
    "hmv": {
        "a": "h:mm a - h:mm a v",
        "h": "h:mm-h:mm a v",
        "m": "h:mm-h:mm a v"
    },
    "Hmv": {
        "H": "HH:mm-HH:mm 'Uhr' v",
        "m": "HH:mm-HH:mm 'Uhr' v"
    },
    "hv": {
        "a": "h a - h a v",
        "h": "h-h a v"
    },
    "Hv": {
        "H": "HH-HH 'Uhr' v"
    },
    "M": {
        "M": "M.-M."
    },
    "Md": {
        "d": "dd.MM. - dd.MM.",
        "M": "dd.MM. - dd.MM."
    },
    "MEd": {
        "d": "E, dd.MM. - E, dd.MM.",
        "M": "E, dd.MM. - E, dd.MM."
    },
    "MMM": {
        "M": "MMM-MMM"
    },
    "MMMd": {
        "d": "d.-d. MMM",
        "M": "d. MMM - d. MMM"
    },
    "MMMEd": {
        "d": "E, d. - E, d. MMM",
        "M": "E, d. MMM - E, d. MMM"
    },
    "MMMM": {
        "M": "LLLL-LLLL"
    },
    "Y": {
        "Y": "Y-Y"
    },
    "YM": {
        "M": "MM.y - MM.y",
        "Y": "MM.y - MM.y"
    },
    "yMd": {
        "d": "dd.MM.y - dd.MM.y",
        "M": "dd.MM.y - dd.MM.y",
    }

```

```
"y": "dd.MM.y - dd.MM.y"
},
"yMEd": {
    "d": "E, dd.MM.y - E, dd.MM.y",
    "M": "E, dd.MM.y - E, dd.MM.y",
    "Y": "E, dd.MM.y - E, dd.MM.y"
},
"yMMM": {
    "M": "MMM-MMM y",
    "Y": "MMM y - MMM y"
},
"yMMMd": {
    "d": "d.-d. MMM y",
    "M": "d. MMM - d. MMM y",
    "Y": "d. MMM y - d. MMM y"
},
"yMMMED": {
    "d": "E, d. - E, d. MMM y",
    "M": "E, d. MMM - E, d. MMM y",
    "Y": "E, d. MMM y - E, d. MMM y"
},
"yMMMM": {
    "M": "MMMM-MMMM y",
    "Y": "MMMM y - MMMM y"
}
}
```

CA-GREGORIAN.JSX

```
{
    "main";
    {
        "de";
        {
            "identity";
            {
                "version";
                {
                    "_number";
                    "$Revision: 12879 $",
                    "_cldrVersion";
                    "30.0.3";
                }
                "language";
                "de";
            }
        }
        "dates";
        {
            "calendars";
```



```
{
  "gregorian":
  {
    "months":
    {
      "format":
      {
        "abbreviated":
        {
          "1";
          "Jan.",
          "2";
          "Feb.",
          "3";
          "März",
          "4";
          "Apr.",
          "5";
          "Mai",
          "6";
          "Juni",
          "7";
          "Juli",
          "8";
          "Aug.",
          "9";
          "Sep.",
          "10";
          "Okt.",
          "11";
          "Nov.",
          "12";
          "Dez.";
        }
        "narrow":
        {
          "1";
          "J",
          "2";
          "F",
          "3";
          "M",
          "4";
          "A",
          "5";
          "M",
          "6";
          "J",
          "7";
          "J",
          "8";
          "A",
          "9";
          "S",
          "10";
          "O",
          "11";
        }
      }
    }
  }
}
```

```
        "N",
        "12";
        "D";
    }
    "wide";
    {
        "1";
        "Januar",
        "2";
        "Februar",
        "3";
        "März",
        "4";
        "April",
        "5";
        "Mai",
        "6";
        "Juni",
        "7";
        "Juli",
        "8";
        "August",
        "9";
        "September",
        "10";
        "Oktober",
        "11";
        "November",
        "12";
        "Dezember";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
        "Jan",
        "2";
        "Feb",
        "3";
        "Mär",
        "4";
        "Apr",
        "5";
        "Mai",
        "6";
        "Jun",
        "7";
        "Jul",
        "8";
        "Aug",
        "9";
        "Sep",
        "10";
        "Okt",
        "11";
```

```
        "Nov",
        "12";
        "Dez";
    }
    "narrow";
    {
        "1";
        "J",
        "2";
        "F",
        "3";
        "M",
        "4";
        "A",
        "5";
        "M",
        "6";
        "J",
        "7";
        "J",
        "8";
        "A",
        "9";
        "S",
        "10";
        "O",
        "11";
        "N",
        "12";
        "D";
    }
    "wide";
    {
        "1";
        "Januar",
        "2";
        "Februar",
        "3";
        "März",
        "4";
        "April",
        "5";
        "Mai",
        "6";
        "Juni",
        "7";
        "Juli",
        "8";
        "August",
        "9";
        "September",
        "10";
        "Oktober",
        "11";
        "November",
        "12";
        "Dezember";
    }
}
```

```
    }  
  }  
  "days";  
  {  
    "format";  
    {  
      "abbreviated";  
      {  
        "sun";  
        "So.",  
        "mon";  
        "Mo.",  
        "tue";  
        "Di.",  
        "wed";  
        "Mi.",  
        "thu";  
        "Do.",  
        "fri";  
        "Fr.",  
        "sat";  
        "Sa.";  
      }  
      "narrow";  
      {  
        "sun";  
        "S",  
        "mon";  
        "M",  
        "tue";  
        "D",  
        "wed";  
        "M",  
        "thu";  
        "D",  
        "fri";  
        "F",  
        "sat";  
        "S";  
      }  
      "short";  
      {  
        "sun";  
        "So.",  
        "mon";  
        "Mo.",  
        "tue";  
        "Di.",  
        "wed";  
        "Mi.",  
        "thu";  
        "Do.",  
        "fri";  
        "Fr.",  
        "sat";  
        "Sa.";
```

```
}
"wide";
{
  "sun";
  "Sonntag",
  "mon";
  "Montag",
  "tue";
  "Dienstag",
  "wed";
  "Mittwoch",
  "thu";
  "Donnerstag",
  "fri";
  "Freitag",
  "sat";
  "Samstag";
}
}
"stand-alone";
{
  "abbreviated";
  {
    "sun";
    "So",
    "mon";
    "Mo",
    "tue";
    "Di",
    "wed";
    "Mi",
    "thu";
    "Do",
    "fri";
    "Fr",
    "sat";
    "Sa";
  }
  "narrow";
  {
    "sun";
    "S",
    "mon";
    "M",
    "tue";
    "D",
    "wed";
    "M",
    "thu";
    "D",
    "fri";
    "F",
    "sat";
    "S";
  }
  "short";
  {
```

```

        "sun";
        "So.",
        "mon";
        "Mo.",
        "tue";
        "Di.",
        "wed";
        "Mi.",
        "thu";
        "Do.",
        "fri";
        "Fr.",
        "sat";
        "Sa.";
    }
    "wide";
    {
        "sun";
        "Sonntag",
        "mon";
        "Montag",
        "tue";
        "Dienstag",
        "wed";
        "Mittwoch",
        "thu";
        "Donnerstag",
        "fri";
        "Freitag",
        "sat";
        "Samstag";
    }
}
"quarters";
{
    "format";
    {
        "abbreviated";
        {
            "1";
            "Q1",
            "2";
            "Q2",
            "3";
            "Q3",
            "4";
            "Q4";
        }
        "narrow";
        {
            "1";
            "1",
            "2";
            "2",
            "3";
            "3",

```

```

        "4";
        "4";
    }
    "wide";
    {
        "1";
        "1. Quartal",
        "2";
        "2. Quartal",
        "3";
        "3. Quartal",
        "4";
        "4. Quartal";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
        "Q1",
        "2";
        "Q2",
        "3";
        "Q3",
        "4";
        "Q4";
    }
    "narrow";
    {
        "1";
        "1",
        "2";
        "2",
        "3";
        "3",
        "4";
        "4";
    }
    "wide";
    {
        "1";
        "1. Quartal",
        "2";
        "2. Quartal",
        "3";
        "3. Quartal",
        "4";
        "4. Quartal";
    }
}
}
"dayPeriods";
{
    "format";
    {
        "abbreviated";

```

```
{
  "midnight";
  "Mitternacht",
    "am";
  "vorm.",
    "pm";
  "nachm.",
    "morning1";
  "morgens",
    "morning2";
  "vormittags",
    "afternoon1";
  "mittags",
    "afternoon2";
  "nachmittags",
    "evening1";
  "abends",
    "night1";
  "nachts";
}
"narrow";
{
  "midnight";
  "Mitternacht",
    "am";
  "vm.",
    "pm";
  "nm.",
    "morning1";
  "morgens",
    "morning2";
  "vormittags",
    "afternoon1";
  "mittags",
    "afternoon2";
  "nachmittags",
    "evening1";
  "abends",
    "night1";
  "nachts";
}
"wide";
{
  "midnight";
  "Mitternacht",
    "am";
  "vorm.",
    "pm";
  "nachm.",
    "morning1";
  "morgens",
    "morning2";
  "vormittags",
    "afternoon1";
  "mittags",
    "afternoon2";
  "nachmittags",
```



```
        "evening1";
        "abends",
        "night1";
        "nachts";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vorm.",
        "pm";
        "nachm.",
        "morning1";
        "Morgen",
        "morning2";
        "Vormittag",
        "afternoon1";
        "Mittag",
        "afternoon2";
        "Nachmittag",
        "evening1";
        "Abend",
        "night1";
        "Nacht";
    }
    "narrow";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vorm.",
        "pm";
        "nachm.",
        "morning1";
        "Morgen",
        "morning2";
        "Vormittag",
        "afternoon1";
        "Mittag",
        "afternoon2";
        "Nachmittag",
        "evening1";
        "Abend",
        "night1";
        "Nacht";
    }
    "wide";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vorm.",
        "pm";
```

```

        "nachm.",
        "morning1";
    "Morgen",
        "morning2";
    "Vormittag",
        "afternoon1";
    "Mittag",
        "afternoon2";
    "Nachmittag",
        "evening1";
    "Abend",
        "night1";
    "Nacht";
    }
    }
}
"eras";
{
    "eraNames";
    {
        "0";
        "v. Chr.",
            "0-alt-variant";
        "vor unserer Zeitrechnung",
            "1";
        "n. Chr.",
            "1-alt-variant";
        "unserer Zeitrechnung";
    }
    "eraAbbr";
    {
        "0";
        "v. Chr.",
            "0-alt-variant";
        "v. u. Z.",
            "1";
        "n. Chr.",
            "1-alt-variant";
        "u. Z.";
    }
    "eraNarrow";
    {
        "0";
        "v. Chr.",
            "0-alt-variant";
        "v. u. Z.",
            "1";
        "n. Chr.",
            "1-alt-variant";
        "u. Z.";
    }
}
"dateFormats";
{
    "full";
    "EEEE, d. MMMM y",
    "long";
}

```

```

        "d. MMMM y",
        "medium";
        "dd.MM.y",
        "short";
        "dd.MM.yy";
    }
    "timeFormats";
    {
        "full";
        "HH:mm:ss zzzz",
        "long";
        "HH:mm:ss z",
        "medium";
        "HH:mm:ss",
        "short";
        "HH:mm";
    }
    "dateTimeFormats";
    {
        "full";
        "{1} 'um' {0}",
        "long";
        "{1} 'um' {0}",
        "medium";
        "{1}, {0}",
        "short";
        "{1}, {0}",
        "availableFormats";
        {
            "d";
            "d",
            "E";
            "ccc",
            "Ed";
            "E, d.",
            "Ehm";
            "E h:mm a",
            "EHm";
            "E, HH:mm",
            "Ehms";
            "E, h:mm:ss a",
            "EHms";
            "E, HH:mm:ss",
            "Gy";
            "y G",
            "GyMMM";
            "MMM y G",
            "GyMMMd";
            "d. MMM y G",
            "GyMMMED";
            "E, d. MMM y G",
            "h";
            "h 'Uhr' a",
            "H";
            "HH 'Uhr' ",
            "hm";
            "h:mm a",

```

```

        "Hm";
        "HH:mm",
        "hms";
        "h:mm:ss a",
        "Hms";
        "HH:mm:ss",
        "hmsv";
        "h:mm:ss a v",
        "Hmsv";
        "HH:mm:ss v",
        "hmv";
        "h:mm a v",
        "Hmv";
        "HH:mm v",
        "M";
        "L",
        "Md";
        "d.M.",
        "MEd";
        "E, d.M.",
        "MMd";
        "d.MM.",
        "MMdd";
        "dd.MM.",
        "MMM";
        "LLL",
        "MMMd";
        "d. MMM",
        "MMMEd";
        "E, d. MMM",
        "MMMMd";
        "d. MMMM",
        "MMMMEd";
        "E, d. MMMM",
        "MMMMW";
        "'Woche' W 'im' MMM",
        "MMMMW";
        "'Woche' W 'im' MMM",
        "ms";
        "mm:ss",
        "y";
        "Y",
        "yM";
        "M.y",
        "yMd";
        "d.M.y",
        "yMEd";
        "E, d.M.y",
        "yMM";
        "MM.y",
        "yMMdd";
        "dd.MM.y",
        "yMMM";
        "MMM y",
        "yMMMd";
        "d. MMM y",
        "yMMMEd";

```

```

        "E, d. MMM y",
        "yMMMM";
        "MMMM y",
        "yQQQ";
        "QQQ y",
        "yQQQQ";
        "QQQQ y",
        "yw";
        "'Woche' w 'des' 'Jahres' y",
        "yw";
        "'Woche' w 'des' 'Jahres' y";
    }
    "appendItems";
    {
        "Day";
        "{0} ({2}: {1})",
        "Day-Of-Week";
        "{0} {1}",
        "Era";
        "{1} {0}",
        "Hour";
        "{0} ({2}: {1})",
        "Minute";
        "{0} ({2}: {1})",
        "Month";
        "{0} ({2}: {1})",
        "Quarter";
        "{0} ({2}: {1})",
        "Second";
        "{0} ({2}: {1})",
        "Timezone";
        "{0} {1}",
        "Week";
        "{0} ({2}: {1})",
        "Year";
        "{1} {0}";
    }
    "intervalFormats";
    {
        "intervalFormatFallback";
        "{0} - {1}",
        "d";
        {
            "d";
            "d.-d.";
        }
        "h";
        {
            "a";
            "h 'Uhr' a - h 'Uhr' a",
            "h";
            "h - h 'Uhr' a";
        }
        "H";
        {
            "H";
            "HH-HH 'Uhr'";
        }
    }

```

```
}
"hm";
{
  "a";
  "h:mm a - h:mm a",
  "h";
  "h:mm-h:mm a",
  "m";
  "h:mm-h:mm a";
}
"Hm";
{
  "H";
  "HH:mm-HH:mm 'Uhr'",
  "m";
  "HH:mm-HH:mm 'Uhr'";
}
"hmV";
{
  "a";
  "h:mm a - h:mm a v",
  "h";
  "h:mm-h:mm a v",
  "m";
  "h:mm-h:mm a v";
}
"HmV";
{
  "H";
  "HH:mm-HH:mm 'Uhr' v",
  "m";
  "HH:mm-HH:mm 'Uhr' v";
}
"hv";
{
  "a";
  "h a - h a v",
  "h";
  "h-h a v";
}
"Hv";
{
  "H";
  "HH-HH 'Uhr' v";
}
"M";
{
  "M";
  "M.-M.";
}
"Md";
{
  "d";
  "dd.MM. - dd.MM.",
  "M";
  "dd.MM. - dd.MM.";
}
```

```

"MEd";
{
    "d";
    "E, dd.MM. - E, dd.MM.";
    "M";
    "E, dd.MM. - E, dd.MM.";
}
"MMM";
{
    "M";
    "MMM-MMM";
}
"MMMd";
{
    "d";
    "d.-d. MMM",
    "M";
    "d. MMM - d. MMM";
}
"MMMEd";
{
    "d";
    "E, d. - E, d. MMM",
    "M";
    "E, d. MMM - E, d. MMM";
}
"MMMM";
{
    "M";
    "LLLL-LLLL";
}
"Y";
{
    "Y";
    "Y-Y";
}
"YM";
{
    "M";
    "MM.Y - MM.Y",
    "Y";
    "MM.Y - MM.Y";
}
"yMd";
{
    "d";
    "dd.MM.Y - dd.MM.Y",
    "M";
    "dd.MM.Y - dd.MM.Y",
    "Y";
    "dd.MM.Y - dd.MM.Y";
}
"yMEd";
{
    "d";
    "E, dd.MM.Y - E, dd.MM.Y",
    "M";

```

```

    "E, dd.MM.y - E, dd.MM.y",
        "Y";
    "E, dd.MM.y - E, dd.MM.y";
}
"yMMM";
{
    "M";
    "MMM-MMM y",
        "Y";
    "MMM y - MMM y";
}
"yMMMd";
{
    "d";
    "d.-d. MMM y",
        "M";
    "d. MMM - d. MMM y",
        "Y";
    "d. MMM y - d. MMM y";
}
"yMMMed";
{
    "d";
    "E, d. - E, d. MMM y",
        "M";
    "E, d. MMM - E, d. MMM y",
        "Y";
    "E, d. MMM y - E, d. MMM y";
}
"yMMMM";
{
    "M";
    "MMMM-MMMM y",
        "Y";
    "MMMM y - MMMM y";
}
}
}
}
}
}
}
}
}

```

CURRENCIES.JSON

```
{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31"
        },
        "language": "de"
      }
    }
  }
}
```



```

    },
    "numbers": {
      "currencies": {
        "ADP": {
          "displayName": "Andorranische Pesete",
          "displayName-count-one": "Andorranische Pesete",
          "displayName-count-other": "Andorranische Peseten",
          "symbol": "ADP"
        },
        "AED": {
          "displayName": "VAE-Dirham",
          "displayName-count-one": "VAE-Dirham",
          "displayName-count-other": "VAE-Dirham",
          "symbol": "AED"
        },
        "AFA": {
          "displayName": "Afghanische Afghani (1927-2002)",
          "displayName-count-one": "Afghanische Afghani (1927-2002)",
          "displayName-count-other": "Afghanische Afghani (1927-2002)",
          "symbol": "AFA"
        },
        "AFN": {
          "displayName": "Afghanischer Afghani",
          "displayName-count-one": "Afghanischer Afghani",
          "displayName-count-other": "Afghanische Afghani",
          "symbol": "AFN"
        },
        "ALK": {
          "displayName": "Albanischer Lek (1946-1965)",
          "displayName-count-one": "Albanischer Lek (1946-1965)",
          "displayName-count-other": "Albanische Lek (1946-1965)"
        },
        "ALL": {
          "displayName": "Albanischer Lek",
          "displayName-count-one": "Albanischer Lek",
          "displayName-count-other": "Albanische Lek",
          "symbol": "ALL"
        },
        "AMD": {
          "displayName": "Armenischer Dram",
          "displayName-count-one": "Armenischer Dram",
          "displayName-count-other": "Armenische Dram",
          "symbol": "AMD"
        },
        "ANG": {
          "displayName": "Niederländische-Antillen-Gulden",
          "displayName-count-one": "Niederländische-Antillen-Gulden",
          "displayName-count-other": "Niederländische-Antillen-Gulden",
          "symbol": "ANG"
        },
        "AOA": {
          "displayName": "Angolanischer Kwanza",
          "displayName-count-one": "Angolanischer Kwanza",
          "displayName-count-other": "Angolanische Kwanza",
          "symbol": "AOA",
          "symbol-alt-narrow": "Kz"
        }
      }
    }
  },

```

```

    "AOK": {
      "displayName": "Angolanischer Kwanza (1977-1990)",
      "displayName-count-one": "Angolanischer Kwanza (1977-1990)",
      "displayName-count-other": "Angolanische Kwanza (1977-1990)",
      "symbol": "AOK"
    },
    "AON": {
      "displayName": "Angolanischer Neuer Kwanza (1990-2000)",
      "displayName-count-one": "Angolanischer Neuer Kwanza (1990-2000)",
      "displayName-count-other": "Angolanische Neue Kwanza (1990-2000)",
      "symbol": "AON"
    },
    "AOR": {
      "displayName": "Angolanischer Kwanza Reajustado (1995-1999)",
      "displayName-count-one": "Angolanischer Kwanza Reajustado (1995-1999)",
      "displayName-count-other": "Angolanische Kwanza Reajustado (1995-1999)",
      "symbol": "AOR"
    },
    "ARA": {
      "displayName": "Argentinischer Austral",
      "displayName-count-one": "Argentinischer Austral",
      "displayName-count-other": "Argentinische Austral",
      "symbol": "ARA"
    },
    "ARL": {
      "displayName": "Argentinischer Peso Ley (1970-1983)",
      "displayName-count-one": "Argentinischer Peso Ley (1970-1983)",
      "displayName-count-other": "Argentinische Pesos Ley (1970-1983)",
      "symbol": "ARL"
    },
    "ARM": {
      "displayName": "Argentinischer Peso (1881-1970)",
      "displayName-count-one": "Argentinischer Peso (1881-1970)",
      "displayName-count-other": "Argentinische Pesos (1881-1970)",
      "symbol": "ARM"
    },
    "ARP": {
      "displayName": "Argentinischer Peso (1983-1985)",
      "displayName-count-one": "Argentinischer Peso (1983-1985)",
      "displayName-count-other": "Argentinische Peso (1983-1985)",
      "symbol": "ARP"
    },
    "ARS": {
      "displayName": "Argentinischer Peso",
      "displayName-count-one": "Argentinischer Peso",
      "displayName-count-other": "Argentinische Pesos",
      "symbol": "ARS",
      "symbol-alt-narrow": "$"
    },
    "ATS": {
      "displayName": "Österreichischer Schilling",
      "displayName-count-one": "Österreichischer Schilling",
      "displayName-count-other": "Österreichische Schilling",

```

```

        "symbol": "öS"
    },
    "AUD": {
        "displayName": "Australischer Dollar",
        "displayName-count-one": "Australischer Dollar",
        "displayName-count-other": "Australische Dollar",
        "symbol": "AU$",
        "symbol-alt-narrow": "$"
    },
    "AWG": {
        "displayName": "Aruba-Florin",
        "displayName-count-one": "Aruba-Florin",
        "displayName-count-other": "Aruba-Florin",
        "symbol": "AWG"
    },
    "AZM": {
        "displayName": "Aserbaidtschan-Manat (1993-2006)",
        "displayName-count-one": "Aserbaidtschan-Manat (1993-2006)",
        "displayName-count-other": "Aserbaidtschan-Manat (1993-2006)",
        "symbol": "AZM"
    },
    "AZN": {
        "displayName": "Aserbaidtschan-Manat",
        "displayName-count-one": "Aserbaidtschan-Manat",
        "displayName-count-other": "Aserbaidtschan-Manat",
        "symbol": "AZN"
    },
    "BAD": {
        "displayName": "Bosnien und Herzegowina Dinar (1992-1994)",
        "displayName-count-one": "Bosnien und Herzegowina Dinar (1992-1994)",
        "displayName-count-other": "Bosnien und Herzegowina Dinar (1992-1994)",
        "symbol": "BAD"
    },
    "BAM": {
        "displayName": "Bosnien und Herzegowina Konvertierbare Mark",
        "displayName-count-one": "Bosnien und Herzegowina Konvertierbare Mark",
        "displayName-count-other": "Bosnien und Herzegowina Konvertierbare Mark",
        "symbol": "BAM",
        "symbol-alt-narrow": "KM"
    },
    "BAN": {
        "displayName": "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
        "displayName-count-one": "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
        "displayName-count-other": "Bosnien und Herzegowina Neue Dinar (1994-1997)",
        "symbol": "BAN"
    },
    "BBD": {
        "displayName": "Barbados-Dollar",
        "displayName-count-one": "Barbados-Dollar",
        "displayName-count-other": "Barbados-Dollar",
        "symbol": "BBD",

```

```

    "symbol-alt-narrow": "$"
  },
  "BDT": {
    "displayName": "Bangladesch-Taka",
    "displayName-count-one": "Bangladesch-Taka",
    "displayName-count-other": "Bangladesch-Taka",
    "symbol": "BDT",
    "symbol-alt-narrow": "৳"
  },
  "BEC": {
    "displayName": "Belgischer Franc (konvertibel)",
    "displayName-count-one": "Belgischer Franc (konvertibel)",
    "displayName-count-other": "Belgische Franc (konvertibel)",
    "symbol": "BEC"
  },
  "BEF": {
    "displayName": "Belgischer Franc",
    "displayName-count-one": "Belgischer Franc",
    "displayName-count-other": "Belgische Franc",
    "symbol": "BEF"
  },
  "BEL": {
    "displayName": "Belgischer Finanz-Franc",
    "displayName-count-one": "Belgischer Finanz-Franc",
    "displayName-count-other": "Belgische Finanz-Franc",
    "symbol": "BEL"
  },
  "BGL": {
    "displayName": "Bulgarische Lew (1962-1999)",
    "displayName-count-one": "Bulgarische Lew (1962-1999)",
    "displayName-count-other": "Bulgarische Lew (1962-1999)",
    "symbol": "BGL"
  },
  "BGM": {
    "displayName": "Bulgarischer Lew (1952-1962)",
    "displayName-count-one": "Bulgarischer Lew (1952-1962)",
    "displayName-count-other": "Bulgarische Lew (1952-1962)",
    "symbol": "BGK"
  },
  "BGN": {
    "displayName": "Bulgarischer Lew",
    "displayName-count-one": "Bulgarischer Lew",
    "displayName-count-other": "Bulgarische Lew",
    "symbol": "BGN"
  },
  "BGO": {
    "displayName": "Bulgarischer Lew (1879-1952)",
    "displayName-count-one": "Bulgarischer Lew (1879-1952)",
    "displayName-count-other": "Bulgarische Lew (1879-1952)",
    "symbol": "BGJ"
  },
  "BHD": {
    "displayName": "Bahrain-Dinar",
    "displayName-count-one": "Bahrain-Dinar",
    "displayName-count-other": "Bahrain-Dinar",
    "symbol": "BHD"
  }

```

```

    },
    "BIF": {
      "displayName": "Burundi-Franc",
      "displayName-count-one": "Burundi-Franc",
      "displayName-count-other": "Burundi-Francs",
      "symbol": "BIF"
    },
    "BMD": {
      "displayName": "Bermuda-Dollar",
      "displayName-count-one": "Bermuda-Dollar",
      "displayName-count-other": "Bermuda-Dollar",
      "symbol": "BMD",
      "symbol-alt-narrow": "$"
    },
    "BND": {
      "displayName": "Brunei-Dollar",
      "displayName-count-one": "Brunei-Dollar",
      "displayName-count-other": "Brunei-Dollar",
      "symbol": "BND",
      "symbol-alt-narrow": "$"
    },
    "BOB": {
      "displayName": "Bolivanischer Boliviano",
      "displayName-count-one": "Bolivanischer Boliviano",
      "displayName-count-other": "Bolivianische Bolivianos",
      "symbol": "BOB",
      "symbol-alt-narrow": "Bs"
    },
    "BOL": {
      "displayName": "Bolivianischer Boliviano (1863-1963)",
      "displayName-count-one": "Bolivianischer Boliviano (1863-1963)",
      "displayName-count-other": "Bolivianische Bolivianos (1863-
1963)",
      "symbol": "BOL"
    },
    "BOP": {
      "displayName": "Bolivianischer Peso",
      "displayName-count-one": "Bolivianischer Peso",
      "displayName-count-other": "Bolivianische Peso",
      "symbol": "BOP"
    },
    "BOV": {
      "displayName": "Boliviansiche Mvdol",
      "displayName-count-one": "Boliviansiche Mvdol",
      "displayName-count-other": "Bolivianische Mvdol",
      "symbol": "BOV"
    },
    "BRB": {
      "displayName": "Brasilianischer Cruzeiro Novo (1967-1986)",
      "displayName-count-one": "Brasilianischer Cruzeiro Novo (1967-
1986)",
      "displayName-count-other": "Brasilianische Cruzeiro Novo (1967-
1986)",
      "symbol": "BRB"
    },
    "BRC": {
      "displayName": "Brasilianischer Cruzado (1986-1989)",

```

```

        "displayName-count-one": "Brasilianischer Cruzado (1986-1989)",
        "displayName-count-other": "Brasilianische Cruzado (1986-1989)",
        "symbol": "BRC"
    },
    "BRE": {
        "displayName": "Brasilianischer Cruzeiro (1990-1993)",
        "displayName-count-one": "Brasilianischer Cruzeiro (1990-1993)",
        "displayName-count-other": "Brasilianische Cruzeiro (1990-1993)",
        "symbol": "BRE"
    },
    "BRL": {
        "displayName": "Brasilianischer Real",
        "displayName-count-one": "Brasilianischer Real",
        "displayName-count-other": "Brasilianische Real",
        "symbol": "R$",
        "symbol-alt-narrow": "R$"
    },
    "BRN": {
        "displayName": "Brasilianischer Cruzado Novo (1989-1990)",
        "displayName-count-one": "Brasilianischer Cruzado Novo (1989-
1990)",
        "displayName-count-other": "Brasilianische Cruzado Novo (1989-
1990)",
        "symbol": "BRN"
    },
    "BRR": {
        "displayName": "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-one": "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-other": "Brasilianische Cruzeiro (1993-1994)",
        "symbol": "BRR"
    },
    "BRZ": {
        "displayName": "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-one": "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-other": "Brasilianischer Cruzeiro (1942-
1967)",
        "symbol": "BRZ"
    },
    "BSD": {
        "displayName": "Bahamas-Dollar",
        "displayName-count-one": "Bahamas-Dollar",
        "displayName-count-other": "Bahamas-Dollar",
        "symbol": "BSD",
        "symbol-alt-narrow": "$"
    },
    "BTN": {
        "displayName": "Bhutan-Ngultrum",
        "displayName-count-one": "Bhutan-Ngultrum",
        "displayName-count-other": "Bhutan-Ngultrum",
        "symbol": "BTN"
    },
    "BUK": {
        "displayName": "Birmanischer Kyat",
        "displayName-count-one": "Birmanischer Kyat",
        "displayName-count-other": "Birmanische Kyat",
        "symbol": "BUK"
    },
    },

```

```
"BWP": {
  "displayName": "Botswanischer Pula",
  "displayName-count-one": "Botswanischer Pula",
  "displayName-count-other": "Botswanische Pula",
  "symbol": "BWP",
  "symbol-alt-narrow": "P"
},
"BYB": {
  "displayName": "Belarus-Rubel (1994-1999)",
  "displayName-count-one": "Belarus-Rubel (1994-1999)",
  "displayName-count-other": "Belarus-Rubel (1994-1999)",
  "symbol": "BYB"
},
"BYN": {
  "displayName": "Weißrussischer Rubel",
  "displayName-count-one": "Weißrussischer Rubel",
  "displayName-count-other": "Weißrussische Rubel",
  "symbol": "BYN",
  "symbol-alt-narrow": "p."
},
"BYR": {
  "displayName": "Weißrussischer Rubel (2000-2016)",
  "displayName-count-one": "Weißrussischer Rubel (2000-2016)",
  "displayName-count-other": "Weißrussische Rubel (2000-2016)",
  "symbol": "BYR"
},
"BZD": {
  "displayName": "Belize-Dollar",
  "displayName-count-one": "Belize-Dollar",
  "displayName-count-other": "Belize-Dollar",
  "symbol": "BZD",
  "symbol-alt-narrow": "$"
},
"CAD": {
  "displayName": "Kanadischer Dollar",
  "displayName-count-one": "Kanadischer Dollar",
  "displayName-count-other": "Kanadische Dollar",
  "symbol": "CA$",
  "symbol-alt-narrow": "$"
},
"CDF": {
  "displayName": "Kongo-Franc",
  "displayName-count-one": "Kongo-Franc",
  "displayName-count-other": "Kongo-Francs",
  "symbol": "CDF"
},
"CHE": {
  "displayName": "WIR-Euro",
  "displayName-count-one": "WIR-Euro",
  "displayName-count-other": "WIR-Euro",
  "symbol": "CHE"
},
"CHF": {
  "displayName": "Schweizer Franken",
  "displayName-count-one": "Schweizer Franken",
  "displayName-count-other": "Schweizer Franken",
  "symbol": "CHF"
}
```

```

    },
    "CHW": {
      "displayName": "WIR Franken",
      "displayName-count-one": "WIR Franken",
      "displayName-count-other": "WIR Franken",
      "symbol": "CHW"
    },
    "CLE": {
      "displayName": "Chilenischer Escudo",
      "displayName-count-one": "Chilenischer Escudo",
      "displayName-count-other": "Chilenische Escudo",
      "symbol": "CLE"
    },
    "CLF": {
      "displayName": "Chilenische Unidades de Fomento",
      "displayName-count-one": "Chilenische Unidades de Fomento",
      "displayName-count-other": "Chilenische Unidades de Fomento",
      "symbol": "CLF"
    },
    "CLP": {
      "displayName": "Chilenischer Peso",
      "displayName-count-one": "Chilenischer Peso",
      "displayName-count-other": "Chilenische Pesos",
      "symbol": "CLP",
      "symbol-alt-narrow": "$"
    },
    "CNX": {
      "displayName": "Dollar der Chinesischen Volksbank",
      "displayName-count-one": "Dollar der Chinesischen Volksbank",
      "displayName-count-other": "Dollar der Chinesischen Volksbank",
      "symbol": "CNX"
    },
    "CNY": {
      "displayName": "Renminbi Yuan",
      "displayName-count-one": "Chinesischer Yuan",
      "displayName-count-other": "Renminbi Yuan",
      "symbol": "CN¥",
      "symbol-alt-narrow": "¥"
    },
    "COP": {
      "displayName": "Kolumbianischer Peso",
      "displayName-count-one": "Kolumbianischer Peso",
      "displayName-count-other": "Kolumbianische Pesos",
      "symbol": "COP",
      "symbol-alt-narrow": "$"
    },
    "COU": {
      "displayName": "Kolumbianische Unidades de valor real",
      "displayName-count-one": "Kolumbianische Unidad de valor real",
      "displayName-count-other": "Kolumbianische Unidades de valor
real",
      "symbol": "COU"
    },
    "CRC": {
      "displayName": "Costa-Rica-Colón",
      "displayName-count-one": "Costa-Rica-Colón",
      "displayName-count-other": "Costa-Rica-Colón",

```



```

        "symbol": "CRC",
        "symbol-alt-narrow": "₡"
    },
    "CSD": {
        "displayName": "Serbischer Dinar (2002-2006)",
        "displayName-count-one": "Serbischer Dinar (2002-2006)",
        "displayName-count-other": "Serbische Dinar (2002-2006)",
        "symbol": "CSD"
    },
    "CSK": {
        "displayName": "Tschechoslowakische Krone",
        "displayName-count-one": "Tschechoslowakische Kronen",
        "displayName-count-other": "Tschechoslowakische Kronen",
        "symbol": "CSK"
    },
    "CUC": {
        "displayName": "Kubanischer Peso (konvertibel)",
        "displayName-count-one": "Kubanischer Peso (konvertibel)",
        "displayName-count-other": "Kubanische Pesos (konvertibel)",
        "symbol": "CUC",
        "symbol-alt-narrow": "Cub$"
    },
    "CUP": {
        "displayName": "Kubanischer Peso",
        "displayName-count-one": "Kubanischer Peso",
        "displayName-count-other": "Kubanische Pesos",
        "symbol": "CUP",
        "symbol-alt-narrow": "$"
    },
    "CVE": {
        "displayName": "Cabo-Verde-Escudo",
        "displayName-count-one": "Cabo-Verde-Escudo",
        "displayName-count-other": "Cabo-Verde-Escudos",
        "symbol": "CVE"
    },
    "CYP": {
        "displayName": "Zypern-Pfund",
        "displayName-count-one": "Zypern Pfund",
        "displayName-count-other": "Zypern Pfund",
        "symbol": "CYP"
    },
    "CZK": {
        "displayName": "Tschechische Krone",
        "displayName-count-one": "Tschechische Krone",
        "displayName-count-other": "Tschechische Kronen",
        "symbol": "CZK",
        "symbol-alt-narrow": "Kč"
    },
    "DDM": {
        "displayName": "Mark der DDR",
        "displayName-count-one": "Mark der DDR",
        "displayName-count-other": "Mark der DDR",
        "symbol": "DDM"
    },
    "DEM": {
        "displayName": "Deutsche Mark",
        "displayName-count-one": "Deutsche Mark",

```

```

        "displayName-count-other": "Deutsche Mark",
        "symbol": "DM"
    },
    "DJF": {
        "displayName": "Dschibuti-Franc",
        "displayName-count-one": "Dschibuti-Franc",
        "displayName-count-other": "Dschibuti-Franc",
        "symbol": "DJF"
    },
    "DKK": {
        "displayName": "Dänische Krone",
        "displayName-count-one": "Dänische Krone",
        "displayName-count-other": "Dänische Kronen",
        "symbol": "DKK",
        "symbol-alt-narrow": "kr"
    },
    "DOP": {
        "displayName": "Dominikanischer Peso",
        "displayName-count-one": "Dominikanischer Peso",
        "displayName-count-other": "Dominikanische Pesos",
        "symbol": "DOP",
        "symbol-alt-narrow": "$"
    },
    "DZD": {
        "displayName": "Algerischer Dinar",
        "displayName-count-one": "Algerischer Dinar",
        "displayName-count-other": "Algerische Dinar",
        "symbol": "DZD"
    },
    "ECS": {
        "displayName": "Ecuadorianischer Sucre",
        "displayName-count-one": "Ecuadorianischer Sucre",
        "displayName-count-other": "Ecuadorianische Sucre",
        "symbol": "ECS"
    },
    "ECV": {
        "displayName": "Verrechnungseinheit für Ecuador",
        "displayName-count-one": "Verrechnungseinheiten für Ecuador",
        "displayName-count-other": "Verrechnungseinheiten für Ecuador",
        "symbol": "ECV"
    },
    "EEK": {
        "displayName": "Estnische Krone",
        "displayName-count-one": "Estnische Krone",
        "displayName-count-other": "Estnische Kronen",
        "symbol": "EEK"
    },
    "EGP": {
        "displayName": "Ägyptisches Pfund",
        "displayName-count-one": "Ägyptisches Pfund",
        "displayName-count-other": "Ägyptische Pfund",
        "symbol": "EGP",
        "symbol-alt-narrow": "E£"
    },
    "ERN": {
        "displayName": "Eritreischer Nakfa",
        "displayName-count-one": "Eritreischer Nakfa",

```

```

        "displayName-count-other": "Eritreische Nakfa",
        "symbol": "ERN"
    },
    "ESA": {
        "displayName": "Spanische Peseta (A-Konten)",
        "displayName-count-one": "Spanische Peseta (A-Konten)",
        "displayName-count-other": "Spanische Peseten (A-Konten)",
        "symbol": "ESA"
    },
    "ESB": {
        "displayName": "Spanische Peseta (konvertibel)",
        "displayName-count-one": "Spanische Peseta (konvertibel)",
        "displayName-count-other": "Spanische Peseten (konvertibel)",
        "symbol": "ESB"
    },
    "ESP": {
        "displayName": "Spanische Peseta",
        "displayName-count-one": "Spanische Peseta",
        "displayName-count-other": "Spanische Peseten",
        "symbol": "ESP",
        "symbol-alt-narrow": "₧"
    },
    "ETB": {
        "displayName": "Äthiopischer Birr",
        "displayName-count-one": "Äthiopischer Birr",
        "displayName-count-other": "Äthiopische Birr",
        "symbol": "ETB"
    },
    "EUR": {
        "displayName": "Euro",
        "displayName-count-one": "Euro",
        "displayName-count-other": "Euro",
        "symbol": "€",
        "symbol-alt-narrow": "€"
    },
    "FIM": {
        "displayName": "Finnische Mark",
        "displayName-count-one": "Finnische Mark",
        "displayName-count-other": "Finnische Mark",
        "symbol": "FIM"
    },
    "FJD": {
        "displayName": "Fidschi-Dollar",
        "displayName-count-one": "Fidschi-Dollar",
        "displayName-count-other": "Fidschi-Dollar",
        "symbol": "FJD",
        "symbol-alt-narrow": "$"
    },
    "FKP": {
        "displayName": "Falkland-Pfund",
        "displayName-count-one": "Falkland-Pfund",
        "displayName-count-other": "Falkland-Pfund",
        "symbol": "FKP",
        "symbol-alt-narrow": "Fl£"
    },
    "FRF": {
        "displayName": "Französischer Franc",

```

```

    "displayName-count-one": "Französischer Franc",
    "displayName-count-other": "Französische Franc",
    "symbol": "FRF"
  },
  "GBP": {
    "displayName": "Britisches Pfund",
    "displayName-count-one": "Britisches Pfund",
    "displayName-count-other": "Britische Pfund",
    "symbol": "£",
    "symbol-alt-narrow": "£"
  },
  "GEK": {
    "displayName": "Georgischer Kupon Larit",
    "displayName-count-one": "Georgischer Kupon Larit",
    "displayName-count-other": "Georgische Kupon Larit",
    "symbol": "GEK"
  },
  "GEL": {
    "displayName": "Georgischer Lari",
    "displayName-count-one": "Georgischer Lari",
    "displayName-count-other": "Georgische Lari",
    "symbol": "GEL",
    "symbol-alt-narrow": "ლ",
    "symbol-alt-variant": "ლ"
  },
  "GHC": {
    "displayName": "Ghanaischer Cedi (1979-2007)",
    "displayName-count-one": "Ghanaischer Cedi (1979-2007)",
    "displayName-count-other": "Ghanaische Cedi (1979-2007)",
    "symbol": "GHC"
  },
  "GHS": {
    "displayName": "Ghanaischer Cedi",
    "displayName-count-one": "Ghanaischer Cedi",
    "displayName-count-other": "Ghanaische Cedi",
    "symbol": "GHS"
  },
  "GIP": {
    "displayName": "Gibraltar-Pfund",
    "displayName-count-one": "Gibraltar-Pfund",
    "displayName-count-other": "Gibraltar Pfund",
    "symbol": "GIP",
    "symbol-alt-narrow": "£"
  },
  "GMD": {
    "displayName": "Gambia-Dalasi",
    "displayName-count-one": "Gambia-Dalasi",
    "displayName-count-other": "Gambia-Dalasi",
    "symbol": "GMD"
  },
  "GNF": {
    "displayName": "Guinea-Franc",
    "displayName-count-one": "Guinea-Franc",
    "displayName-count-other": "Guinea-Franc",
    "symbol": "GNF",
    "symbol-alt-narrow": "F.G."
  },
  },

```

```
"GNS": {
  "displayName": "Guineischer Syli",
  "displayName-count-one": "Guineischer Syli",
  "displayName-count-other": "Guineische Syli",
  "symbol": "GNS"
},
"GQE": {
  "displayName": "Äquatorialguinea-Ekwele",
  "displayName-count-one": "Äquatorialguinea-Ekwele",
  "displayName-count-other": "Äquatorialguinea-Ekwele",
  "symbol": "GQE"
},
"GRD": {
  "displayName": "Griechische Drachme",
  "displayName-count-one": "Griechische Drachme",
  "displayName-count-other": "Griechische Drachmen",
  "symbol": "GRD"
},
"GTQ": {
  "displayName": "Guatemaltekischer Quetzal",
  "displayName-count-one": "Guatemaltekischer Quetzal",
  "displayName-count-other": "Guatemaltekische Quetzales",
  "symbol": "GTQ",
  "symbol-alt-narrow": "Q"
},
"GWE": {
  "displayName": "Portugiesisch Guinea Escudo",
  "displayName-count-one": "Portugiesisch Guinea Escudo",
  "displayName-count-other": "Portugiesisch Guinea Escudo",
  "symbol": "GWE"
},
"GWP": {
  "displayName": "Guinea-Bissau Peso",
  "displayName-count-one": "Guinea-Bissau Peso",
  "displayName-count-other": "Guinea-Bissau Pesos",
  "symbol": "GWP"
},
"GYD": {
  "displayName": "Guyana-Dollar",
  "displayName-count-one": "Guyana-Dollar",
  "displayName-count-other": "Guyana-Dollar",
  "symbol": "GYD",
  "symbol-alt-narrow": "$"
},
"HKD": {
  "displayName": "Hongkong-Dollar",
  "displayName-count-one": "Hongkong-Dollar",
  "displayName-count-other": "Hongkong-Dollar",
  "symbol": "HK$",
  "symbol-alt-narrow": "$"
},
"HNL": {
  "displayName": "Honduras-Lempira",
  "displayName-count-one": "Honduras-Lempira",
  "displayName-count-other": "Honduras-Lempira",
  "symbol": "HNL",
  "symbol-alt-narrow": "L"
```

```
    },
    "HRD": {
      "displayName": "Kroatischer Dinar",
      "displayName-count-one": "Kroatischer Dinar",
      "displayName-count-other": "Kroatische Dinar",
      "symbol": "HRD"
    },
    "HRK": {
      "displayName": "Kroatischer Kuna",
      "displayName-count-one": "Kroatischer Kuna",
      "displayName-count-other": "Kroatische Kuna",
      "symbol": "HRK",
      "symbol-alt-narrow": "kn"
    },
    "HTG": {
      "displayName": "Haitianische Gourde",
      "displayName-count-one": "Haitianische Gourde",
      "displayName-count-other": "Haitianische Gourdes",
      "symbol": "HTG"
    },
    "HUF": {
      "displayName": "Ungarischer Forint",
      "displayName-count-one": "Ungarischer Forint",
      "displayName-count-other": "Ungarische Forint",
      "symbol": "HUF",
      "symbol-alt-narrow": "Ft"
    },
    "IDR": {
      "displayName": "Indonesische Rupiah",
      "displayName-count-one": "Indonesische Rupiah",
      "displayName-count-other": "Indonesische Rupiah",
      "symbol": "IDR",
      "symbol-alt-narrow": "Rp"
    },
    "IEP": {
      "displayName": "Irisches Pfund",
      "displayName-count-one": "Irisches Pfund",
      "displayName-count-other": "Irische Pfund",
      "symbol": "IEP"
    },
    "ILP": {
      "displayName": "Israelisches Pfund",
      "displayName-count-one": "Israelisches Pfund",
      "displayName-count-other": "Israelische Pfund",
      "symbol": "ILP"
    },
    "ILR": {
      "displayName": "Israelischer Schekel (1980-1985)",
      "displayName-count-one": "Israelischer Schekel (1980-1985)",
      "displayName-count-other": "Israelische Schekel (1980-1985)"
    },
    "ILS": {
      "displayName": "Israelischer Neuer Schekel",
      "displayName-count-one": "Israelischer Neuer Schekel",
      "displayName-count-other": "Israelische Neue Schekel",
      "symbol": "₪",
      "symbol-alt-narrow": "₪"
```

```

    },
    "INR": {
      "displayName": "Indische Rupie",
      "displayName-count-one": "Indische Rupie",
      "displayName-count-other": "Indische Rupien",
      "symbol": "₹",
      "symbol-alt-narrow": "₹"
    },
    "IQD": {
      "displayName": "Irakischer Dinar",
      "displayName-count-one": "Irakischer Dinar",
      "displayName-count-other": "Irakische Dinar",
      "symbol": "IQD"
    },
    "IRR": {
      "displayName": "Iranischer Rial",
      "displayName-count-one": "Iranischer Rial",
      "displayName-count-other": "Iranische Rial",
      "symbol": "IRR"
    },
    "ISJ": {
      "displayName": "Isländische Krone (1918-1981)",
      "displayName-count-one": "Isländische Krone (1918-1981)",
      "displayName-count-other": "Isländische Kronen (1918-1981)"
    },
    "ISK": {
      "displayName": "Isländische Krone",
      "displayName-count-one": "Isländische Krone",
      "displayName-count-other": "Isländische Kronen",
      "symbol": "ISK",
      "symbol-alt-narrow": "kr"
    },
    "ITL": {
      "displayName": "Italienische Lira",
      "displayName-count-one": "Italienische Lira",
      "displayName-count-other": "Italienische Lire",
      "symbol": "ITL"
    },
    "JMD": {
      "displayName": "Jamaika-Dollar",
      "displayName-count-one": "Jamaika-Dollar",
      "displayName-count-other": "Jamaika-Dollar",
      "symbol": "JMD",
      "symbol-alt-narrow": "$"
    },
    "JOD": {
      "displayName": "Jordanischer Dinar",
      "displayName-count-one": "Jordanischer Dinar",
      "displayName-count-other": "Jordanische Dinar",
      "symbol": "JOD"
    },
    "JPY": {
      "displayName": "Japanischer Yen",
      "displayName-count-one": "Japanischer Yen",
      "displayName-count-other": "Japanische Yen",
      "symbol": "¥",
      "symbol-alt-narrow": "¥"
    }
  }

```

```

    },
    "KES": {
      "displayName": "Kenia-Schilling",
      "displayName-count-one": "Kenia-Schilling",
      "displayName-count-other": "Kenia-Schilling",
      "symbol": "KES"
    },
    "KGS": {
      "displayName": "Kirgisischer Som",
      "displayName-count-one": "Kirgisischer Som",
      "displayName-count-other": "Kirgisische Som",
      "symbol": "KGS"
    },
    "KHR": {
      "displayName": "Kambodschanischer Riel",
      "displayName-count-one": "Kambodschanischer Riel",
      "displayName-count-other": "Kambodschanische Riel",
      "symbol": "KHR",
      "symbol-alt-narrow": "៛"
    },
    "KMF": {
      "displayName": "Komoren-Franc",
      "displayName-count-one": "Komoren-Franc",
      "displayName-count-other": "Komoren-Francs",
      "symbol": "KMF",
      "symbol-alt-narrow": "FC"
    },
    "KPW": {
      "displayName": "Nordkoreanischer Won",
      "displayName-count-one": "Nordkoreanischer Won",
      "displayName-count-other": "Nordkoreanische Won",
      "symbol": "KPW",
      "symbol-alt-narrow": "₩"
    },
    "KRH": {
      "displayName": "Südkoreanischer Hwan (1953-1962)",
      "displayName-count-one": "Südkoreanischer Hwan (1953-1962)",
      "displayName-count-other": "Südkoreanischer Hwan (1953-1962)",
      "symbol": "KRH"
    },
    "KRO": {
      "displayName": "Südkoreanischer Won (1945-1953)",
      "displayName-count-one": "Südkoreanischer Won (1945-1953)",
      "displayName-count-other": "Südkoreanischer Won (1945-1953)",
      "symbol": "KRO"
    },
    "KRW": {
      "displayName": "Südkoreanischer Won",
      "displayName-count-one": "Südkoreanischer Won",
      "displayName-count-other": "Südkoreanische Won",
      "symbol": "₩",
      "symbol-alt-narrow": "₩"
    },
    "KWD": {
      "displayName": "Kuwait-Dinar",
      "displayName-count-one": "Kuwait-Dinar",

```



```

        "displayName-count-other": "Kuwait-Dinar",
        "symbol": "KWD"
    },
    "KYD": {
        "displayName": "Kaiman-Dollar",
        "displayName-count-one": "Kaiman-Dollar",
        "displayName-count-other": "Kaiman-Dollar",
        "symbol": "KYD",
        "symbol-alt-narrow": "$"
    },
    "KZT": {
        "displayName": "Kasachischer Tenge",
        "displayName-count-one": "Kasachischer Tenge",
        "displayName-count-other": "Kasachische Tenge",
        "symbol": "KZT",
        "symbol-alt-narrow": "T"
    },
    "LAK": {
        "displayName": "Laotischer Kip",
        "displayName-count-one": "Laotischer Kip",
        "displayName-count-other": "Laotische Kip",
        "symbol": "LAK",
        "symbol-alt-narrow": "₭"
    },
    "LBP": {
        "displayName": "Libanesisches Pfund",
        "displayName-count-one": "Libanesisches Pfund",
        "displayName-count-other": "Libanesische Pfund",
        "symbol": "LBP",
        "symbol-alt-narrow": "L₶"
    },
    "LKR": {
        "displayName": "Sri-Lanka-Rupie",
        "displayName-count-one": "Sri-Lanka-Rupie",
        "displayName-count-other": "Sri-Lanka-Rupien",
        "symbol": "LKR",
        "symbol-alt-narrow": "Rs"
    },
    "LRD": {
        "displayName": "Liberianischer Dollar",
        "displayName-count-one": "Liberianischer Dollar",
        "displayName-count-other": "Liberianische Dollar",
        "symbol": "LRD",
        "symbol-alt-narrow": "$"
    },
    "LSL": {
        "displayName": "Loti",
        "displayName-count-one": "Loti",
        "displayName-count-other": "Loti",
        "symbol": "LSL"
    },
    "LTL": {
        "displayName": "Litauischer Litas",
        "displayName-count-one": "Litauischer Litas",
        "displayName-count-other": "Litauische Litas",
        "symbol": "LTL",
        "symbol-alt-narrow": "Lt"
    }

```

```

    },
    "LTT": {
      "displayName": "Litauischer Talonas",
      "displayName-count-one": "Litauische Talonas",
      "displayName-count-other": "Litauische Talonas",
      "symbol": "LTT"
    },
    "LUC": {
      "displayName": "Luxemburgischer Franc (konvertibel)",
      "displayName-count-one": "Luxemburgische Franc (konvertibel)",
      "displayName-count-other": "Luxemburgische Franc (konvertibel)",
      "symbol": "LUC"
    },
    "LUF": {
      "displayName": "Luxemburgischer Franc",
      "displayName-count-one": "Luxemburgische Franc",
      "displayName-count-other": "Luxemburgische Franc",
      "symbol": "LUF"
    },
    "LUL": {
      "displayName": "Luxemburgischer Finanz-Franc",
      "displayName-count-one": "Luxemburgische Finanz-Franc",
      "displayName-count-other": "Luxemburgische Finanz-Franc",
      "symbol": "LUL"
    },
    "LVL": {
      "displayName": "Lettischer Lats",
      "displayName-count-one": "Lettischer Lats",
      "displayName-count-other": "Lettische Lats",
      "symbol": "LVL",
      "symbol-alt-narrow": "Ls"
    },
    "LVR": {
      "displayName": "Lettischer Rubel",
      "displayName-count-one": "Lettische Rubel",
      "displayName-count-other": "Lettische Rubel",
      "symbol": "LVR"
    },
    "LYD": {
      "displayName": "Libyscher Dinar",
      "displayName-count-one": "Libyscher Dinar",
      "displayName-count-other": "Libysche Dinar",
      "symbol": "LYD"
    },
    "MAD": {
      "displayName": "Marokkanischer Dirham",
      "displayName-count-one": "Marokkanischer Dirham",
      "displayName-count-other": "Marokkanische Dirham",
      "symbol": "MAD"
    },
    "MAF": {
      "displayName": "Marokkanischer Franc",
      "displayName-count-one": "Marokkanische Franc",
      "displayName-count-other": "Marokkanische Franc",
      "symbol": "MAF"
    },
    "MCF": {

```

```

        "displayName": "Monegassischer Franc",
        "displayName-count-one": "Monegassischer Franc",
        "displayName-count-other": "Monegassische Franc",
        "symbol": "MCF"
    },
    "MDC": {
        "displayName": "Moldau-Cupon",
        "displayName-count-one": "Moldau-Cupon",
        "displayName-count-other": "Moldau-Cupon",
        "symbol": "MDC"
    },
    "MDL": {
        "displayName": "Moldau-Leu",
        "displayName-count-one": "Moldau-Leu",
        "displayName-count-other": "Moldau-Leu",
        "symbol": "MDL"
    },
    "MGA": {
        "displayName": "Madagaskar-Ariary",
        "displayName-count-one": "Madagaskar-Ariary",
        "displayName-count-other": "Madagaskar-Ariary",
        "symbol": "MGA",
        "symbol-alt-narrow": "Ar"
    },
    "MGF": {
        "displayName": "Madagaskar-Franc",
        "displayName-count-one": "Madagaskar-Franc",
        "displayName-count-other": "Madagaskar-Franc",
        "symbol": "MGF"
    },
    "MKD": {
        "displayName": "Mazedonischer Denar",
        "displayName-count-one": "Mazedonischer Denar",
        "displayName-count-other": "Mazedonische Denari",
        "symbol": "MKD"
    },
    "MKN": {
        "displayName": "Mazedonischer Denar (1992-1993)",
        "displayName-count-one": "Mazedonischer Denar (1992-1993)",
        "displayName-count-other": "Mazedonische Denar (1992-1993)",
        "symbol": "MKN"
    },
    "MLF": {
        "displayName": "Malischer Franc",
        "displayName-count-one": "Malische Franc",
        "displayName-count-other": "Malische Franc",
        "symbol": "MLF"
    },
    "MMK": {
        "displayName": "Myanmarischer Kyat",
        "displayName-count-one": "Myanmarischer Kyat",
        "displayName-count-other": "Myanmarische Kyat",
        "symbol": "MMK",
        "symbol-alt-narrow": "K"
    },
    "MNT": {
        "displayName": "Mongolischer Tögrög",

```

```

        "displayName-count-one": "Mongolischer Tögrög",
        "displayName-count-other": "Mongolische Tögrög",
        "symbol": "MNT",
        "symbol-alt-narrow": "₮"
    },
    "MOP": {
        "displayName": "Macao-Pataca",
        "displayName-count-one": "Macao-Pataca",
        "displayName-count-other": "Macao-Pataca",
        "symbol": "MOP"
    },
    "MRO": {
        "displayName": "Mauretanischer Ouguiya",
        "displayName-count-one": "Mauretanischer Ouguiya",
        "displayName-count-other": "Mauretanische Ouguiya",
        "symbol": "MRO"
    },
    "MTL": {
        "displayName": "Maltesische Lira",
        "displayName-count-one": "Maltesische Lira",
        "displayName-count-other": "Maltesische Lira",
        "symbol": "MTL"
    },
    "MTP": {
        "displayName": "Maltesisches Pfund",
        "displayName-count-one": "Maltesische Pfund",
        "displayName-count-other": "Maltesische Pfund",
        "symbol": "MTP"
    },
    "MUR": {
        "displayName": "Mauritius-Rupie",
        "displayName-count-one": "Mauritius-Rupie",
        "displayName-count-other": "Mauritius-Rupien",
        "symbol": "MUR",
        "symbol-alt-narrow": "Rs"
    },
    "MVP": {
        "displayName": "Malediven-Rupie (alt)",
        "displayName-count-one": "Malediven-Rupie (alt)",
        "displayName-count-other": "Malediven-Rupien (alt)"
    },
    "MVR": {
        "displayName": "Malediven-Rufiyaa",
        "displayName-count-one": "Malediven-Rufiyaa",
        "displayName-count-other": "Malediven-Rupien",
        "symbol": "MVR"
    },
    "MWK": {
        "displayName": "Malawi-Kwacha",
        "displayName-count-one": "Malawi-Kwacha",
        "displayName-count-other": "Malawi-Kwacha",
        "symbol": "MWK"
    },
    "MXN": {
        "displayName": "Mexikanischer Peso",
        "displayName-count-one": "Mexikanischer Peso",
        "displayName-count-other": "Mexikanische Pesos",

```

```

        "symbol": "MX$",
        "symbol-alt-narrow": "$"
    },
    "MXP": {
        "displayName": "Mexikanischer Silber-Peso (1861-1992)",
        "displayName-count-one": "Mexikanische Silber-Peso (1861-1992)",
        "displayName-count-other": "Mexikanische Silber-Pesos (1861-
1992)",
        "symbol": "MXP"
    },
    "MXV": {
        "displayName": "Mexicanischer Unidad de Inversion (UDI)",
        "displayName-count-one": "Mexicanischer Unidad de Inversion
(UDI)",
        "displayName-count-other": "Mexikanische Unidad de Inversion
(UDI)",
        "symbol": "MXV"
    },
    "MYR": {
        "displayName": "Malaysischer Ringgit",
        "displayName-count-one": "Malaysischer Ringgit",
        "displayName-count-other": "Malaysische Ringgit",
        "symbol": "MYR",
        "symbol-alt-narrow": "RM"
    },
    "MZE": {
        "displayName": "Mosambikanischer Escudo",
        "displayName-count-one": "Mozambikanische Escudo",
        "displayName-count-other": "Mozambikanische Escudo",
        "symbol": "MZE"
    },
    "MZM": {
        "displayName": "Mosambikanischer Metical (1980-2006)",
        "displayName-count-one": "Mosambikanischer Metical (1980-2006)",
        "displayName-count-other": "Mosambikanische Meticais (1980-
2006)",
        "symbol": "MZM"
    },
    "MZN": {
        "displayName": "Mosambikanischer Metical",
        "displayName-count-one": "Mosambikanischer Metical",
        "displayName-count-other": "Mosambikanische Meticais",
        "symbol": "MZN"
    },
    "NAD": {
        "displayName": "Namibia-Dollar",
        "displayName-count-one": "Namibia-Dollar",
        "displayName-count-other": "Namibia-Dollar",
        "symbol": "NAD",
        "symbol-alt-narrow": "$"
    },
    "NGN": {
        "displayName": "Nigerianischer Naira",
        "displayName-count-one": "Nigerianischer Naira",
        "displayName-count-other": "Nigerianische Naira",
        "symbol": "NGN",
        "symbol-alt-narrow": "₦"
    }

```

```
    },
    "NIC": {
      "displayName": "Nicaraguanischer Córdoba (1988-1991)",
      "displayName-count-one": "Nicaraguanischer Córdoba (1988-1991)",
      "displayName-count-other": "Nicaraguanische Córdoba (1988-1991)",
      "symbol": "NIC"
    },
    "NIO": {
      "displayName": "Nicaragua-Córdoba",
      "displayName-count-one": "Nicaragua-Córdoba",
      "displayName-count-other": "Nicaragua-Córdobas",
      "symbol": "NIO",
      "symbol-alt-narrow": "C$"
    },
    "NLG": {
      "displayName": "Niederländischer Gulden",
      "displayName-count-one": "Niederländischer Gulden",
      "displayName-count-other": "Niederländische Gulden",
      "symbol": "NLG"
    },
    "NOK": {
      "displayName": "Norwegische Krone",
      "displayName-count-one": "Norwegische Krone",
      "displayName-count-other": "Norwegische Kronen",
      "symbol": "NOK",
      "symbol-alt-narrow": "kr"
    },
    "NPR": {
      "displayName": "Nepalesische Rupie",
      "displayName-count-one": "Nepalesische Rupie",
      "displayName-count-other": "Nepalesische Rupien",
      "symbol": "NPR",
      "symbol-alt-narrow": "Rs"
    },
    "NZD": {
      "displayName": "Neuseeland-Dollar",
      "displayName-count-one": "Neuseeland-Dollar",
      "displayName-count-other": "Neuseeland-Dollar",
      "symbol": "NZ$",
      "symbol-alt-narrow": "$"
    },
    "OMR": {
      "displayName": "Omanischer Rial",
      "displayName-count-one": "Omanischer Rial",
      "displayName-count-other": "Omanische Rials",
      "symbol": "OMR"
    },
    "PAB": {
      "displayName": "Panamaischer Balboa",
      "displayName-count-one": "Panamaischer Balboa",
      "displayName-count-other": "Panamaische Balboas",
      "symbol": "PAB"
    },
    "PEI": {
      "displayName": "Peruanischer Inti",
      "displayName-count-one": "Peruanische Inti",
      "displayName-count-other": "Peruanische Inti",
```

```

        "symbol": "PEI"
    },
    "PEN": {
        "displayName": "Peruanischer Sol",
        "displayName-count-one": "Peruanischer Sol",
        "displayName-count-other": "Peruanische Sol",
        "symbol": "PEN"
    },
    "PES": {
        "displayName": "Peruanischer Sol (1863-1965)",
        "displayName-count-one": "Peruanischer Sol (1863-1965)",
        "displayName-count-other": "Peruanische Sol (1863-1965)",
        "symbol": "PES"
    },
    "PGK": {
        "displayName": "Papua-Neuguineischer Kina",
        "displayName-count-one": "Papua-Neuguineischer Kina",
        "displayName-count-other": "Papua-Neuguineische Kina",
        "symbol": "PGK"
    },
    "PHP": {
        "displayName": "Philippinischer Peso",
        "displayName-count-one": "Philippinischer Peso",
        "displayName-count-other": "Philippinische Pesos",
        "symbol": "PHP",
        "symbol-alt-narrow": "₱"
    },
    "PKR": {
        "displayName": "Pakistanische Rupie",
        "displayName-count-one": "Pakistanische Rupie",
        "displayName-count-other": "Pakistanische Rupien",
        "symbol": "PKR",
        "symbol-alt-narrow": "Rs"
    },
    "PLN": {
        "displayName": "Polnischer Złoty",
        "displayName-count-one": "Polnischer Złoty",
        "displayName-count-other": "Polnische Złoty",
        "symbol": "PLN",
        "symbol-alt-narrow": "zł"
    },
    "PLZ": {
        "displayName": "Polnischer Zloty (1950-1995)",
        "displayName-count-one": "Polnischer Zloty (1950-1995)",
        "displayName-count-other": "Polnische Zloty (1950-1995)",
        "symbol": "PLZ"
    },
    "PTE": {
        "displayName": "Portugiesischer Escudo",
        "displayName-count-one": "Portugiesische Escudo",
        "displayName-count-other": "Portugiesische Escudo",
        "symbol": "PTE"
    },
    "PYG": {
        "displayName": "Paraguayischer Guaraní",
        "displayName-count-one": "Paraguayischer Guaraní",
        "displayName-count-other": "Paraguayische Guaraníes",

```

```

        "symbol": "PYG",
        "symbol-alt-narrow": "₲"
    },
    "QAR": {
        "displayName": "Katar-Riyal",
        "displayName-count-one": "Katar-Riyal",
        "displayName-count-other": "Katar-Riyal",
        "symbol": "QAR"
    },
    "RHD": {
        "displayName": "Rhodesischer Dollar",
        "displayName-count-one": "Rhodesische Dollar",
        "displayName-count-other": "Rhodesische Dollar",
        "symbol": "RHD"
    },
    "ROL": {
        "displayName": "Rumänischer Leu (1952-2006)",
        "displayName-count-one": "Rumänischer Leu (1952-2006)",
        "displayName-count-other": "Rumänische Leu (1952-2006)",
        "symbol": "ROL"
    },
    "RON": {
        "displayName": "Rumänischer Leu",
        "displayName-count-one": "Rumänischer Leu",
        "displayName-count-other": "Rumänische Leu",
        "symbol": "RON",
        "symbol-alt-narrow": "L"
    },
    "RSD": {
        "displayName": "Serbischer Dinar",
        "displayName-count-one": "Serbischer Dinar",
        "displayName-count-other": "Serbische Dinaren",
        "symbol": "RSD"
    },
    "RUB": {
        "displayName": "Russischer Rubel",
        "displayName-count-one": "Russischer Rubel",
        "displayName-count-other": "Russische Rubel",
        "symbol": "RUB",
        "symbol-alt-narrow": "₽"
    },
    "RUR": {
        "displayName": "Russischer Rubel (1991-1998)",
        "displayName-count-one": "Russischer Rubel (1991-1998)",
        "displayName-count-other": "Russische Rubel (1991-1998)",
        "symbol": "RUR",
        "symbol-alt-narrow": "p."
    },
    "RWF": {
        "displayName": "Ruanda-Franc",
        "displayName-count-one": "Ruanda-Franc",
        "displayName-count-other": "Ruanda-Francs",
        "symbol": "RWF",
        "symbol-alt-narrow": "F.Rw"
    },
    "SAR": {
        "displayName": "Saudi-Rial",

```



```

        "displayName-count-one": "Saudi-Rial",
        "displayName-count-other": "Saudi-Rial",
        "symbol": "SAR"
    },
    "SBD": {
        "displayName": "Salomonen-Dollar",
        "displayName-count-one": "Salomonen-Dollar",
        "displayName-count-other": "Salomonen-Dollar",
        "symbol": "SBD",
        "symbol-alt-narrow": "$"
    },
    "SCR": {
        "displayName": "Seychellen-Rupie",
        "displayName-count-one": "Seychellen-Rupie",
        "displayName-count-other": "Seychellen-Rupien",
        "symbol": "SCR"
    },
    "SDD": {
        "displayName": "Sudanesischer Dinar (1992-2007)",
        "displayName-count-one": "Sudanesischer Dinar (1992-2007)",
        "displayName-count-other": "Sudanesische Dinar (1992-2007)",
        "symbol": "SDD"
    },
    "SDG": {
        "displayName": "Sudanesisches Pfund",
        "displayName-count-one": "Sudanesisches Pfund",
        "displayName-count-other": "Sudanesische Pfund",
        "symbol": "SDG"
    },
    "SDP": {
        "displayName": "Sudanesisches Pfund (1957-1998)",
        "displayName-count-one": "Sudanesisches Pfund (1957-1998)",
        "displayName-count-other": "Sudanesische Pfund (1957-1998)",
        "symbol": "SDP"
    },
    "SEK": {
        "displayName": "Schwedische Krone",
        "displayName-count-one": "Schwedische Krone",
        "displayName-count-other": "Schwedische Kronen",
        "symbol": "SEK",
        "symbol-alt-narrow": "kr"
    },
    "SGD": {
        "displayName": "Singapur-Dollar",
        "displayName-count-one": "Singapur-Dollar",
        "displayName-count-other": "Singapur-Dollar",
        "symbol": "SGD",
        "symbol-alt-narrow": "$"
    },
    "SHP": {
        "displayName": "St. Helena-Pfund",
        "displayName-count-one": "St. Helena-Pfund",
        "displayName-count-other": "St. Helena-Pfund",
        "symbol": "SHP",
        "symbol-alt-narrow": "£"
    },
    "SIT": {

```

```

        "displayName": "Slowenischer Tolar",
        "displayName-count-one": "Slowenischer Tolar",
        "displayName-count-other": "Slowenische Tolar",
        "symbol": "SIT"
    },
    "SKK": {
        "displayName": "Slowakische Krone",
        "displayName-count-one": "Slowakische Kronen",
        "displayName-count-other": "Slowakische Kronen",
        "symbol": "SKK"
    },
    "SLL": {
        "displayName": "Sierra-leonischer Leone",
        "displayName-count-one": "Sierra-leonischer Leone",
        "displayName-count-other": "Sierra-leonische Leones",
        "symbol": "SLL"
    },
    "SOS": {
        "displayName": "Somalia-Schilling",
        "displayName-count-one": "Somalia-Schilling",
        "displayName-count-other": "Somalia-Schilling",
        "symbol": "SOS"
    },
    "SRD": {
        "displayName": "Suriname-Dollar",
        "displayName-count-one": "Suriname-Dollar",
        "displayName-count-other": "Suriname-Dollar",
        "symbol": "SRD",
        "symbol-alt-narrow": "$"
    },
    "SRG": {
        "displayName": "Suriname Gulden",
        "displayName-count-one": "Suriname-Gulden",
        "displayName-count-other": "Suriname-Gulden",
        "symbol": "SRG"
    },
    "SSP": {
        "displayName": "Südsudanesisches Pfund",
        "displayName-count-one": "Südsudanesisches Pfund",
        "displayName-count-other": "Südsudanesische Pfund",
        "symbol": "SSP",
        "symbol-alt-narrow": "£"
    },
    "STD": {
        "displayName": "São-toméischer Dobra",
        "displayName-count-one": "São-toméischer Dobra",
        "displayName-count-other": "São-toméische Dobra",
        "symbol": "STD",
        "symbol-alt-narrow": "Db"
    },
    "SUR": {
        "displayName": "Sowjetischer Rubel",
        "displayName-count-one": "Sowjetische Rubel",
        "displayName-count-other": "Sowjetische Rubel",
        "symbol": "SUR"
    },
    "SVC": {

```

```

        "displayName": "El Salvador Colon",
        "displayName-count-one": "El Salvador-Colon",
        "displayName-count-other": "El Salvador-Colon",
        "symbol": "SVC"
    },
    "SYP": {
        "displayName": "Syrisches Pfund",
        "displayName-count-one": "Syrisches Pfund",
        "displayName-count-other": "Syrische Pfund",
        "symbol": "SYP",
        "symbol-alt-narrow": "SYP"
    },
    "SZL": {
        "displayName": "Swasiländischer Lilangeni",
        "displayName-count-one": "Swasiländischer Lilangeni",
        "displayName-count-other": "Swasiländische Emalangeni",
        "symbol": "SZL"
    },
    "THB": {
        "displayName": "Thailändischer Baht",
        "displayName-count-one": "Thailändischer Baht",
        "displayName-count-other": "Thailändische Baht",
        "symbol": "฿",
        "symbol-alt-narrow": "฿"
    },
    "TJR": {
        "displayName": "Tadschikistan Rubel",
        "displayName-count-one": "Tadschikistan-Rubel",
        "displayName-count-other": "Tadschikistan-Rubel",
        "symbol": "TJR"
    },
    "TJS": {
        "displayName": "Tadschikistan-Somoni",
        "displayName-count-one": "Tadschikistan-Somoni",
        "displayName-count-other": "Tadschikistan-Somoni",
        "symbol": "TJS"
    },
    "TMM": {
        "displayName": "Turkmenistan-Manat (1993-2009)",
        "displayName-count-one": "Turkmenistan-Manat (1993-2009)",
        "displayName-count-other": "Turkmenistan-Manat (1993-2009)",
        "symbol": "TMM"
    },
    "TMT": {
        "displayName": "Turkmenistan-Manat",
        "displayName-count-one": "Turkmenistan-Manat",
        "displayName-count-other": "Turkmenistan-Manat",
        "symbol": "TMT"
    },
    "TND": {
        "displayName": "Tunesischer Dinar",
        "displayName-count-one": "Tunesischer Dinar",
        "displayName-count-other": "Tunesische Dinar",
        "symbol": "TND"
    },
    "TOP": {
        "displayName": "Tongaischer Paʻanga",

```

```

        "displayName-count-one": "Tongaischer Pa'anga",
        "displayName-count-other": "Tongaische Pa'anga",
        "symbol": "TOP",
        "symbol-alt-narrow": "T$"
    },
    "TPE": {
        "displayName": "Timor-Escudo",
        "displayName-count-one": "Timor-Escudo",
        "displayName-count-other": "Timor-Escudo",
        "symbol": "TPE"
    },
    "TRL": {
        "displayName": "Türkische Lira (1922-2005)",
        "displayName-count-one": "Türkische Lira (1922-2005)",
        "displayName-count-other": "Türkische Lira (1922-2005)",
        "symbol": "TRL"
    },
    "TRY": {
        "displayName": "Türkische Lira",
        "displayName-count-one": "Türkische Lira",
        "displayName-count-other": "Türkische Lira",
        "symbol": "TRY",
        "symbol-alt-narrow": "₺",
        "symbol-alt-variant": "TL"
    },
    "TTD": {
        "displayName": "Trinidad und Tobago-Dollar",
        "displayName-count-one": "Trinidad und Tobago-Dollar",
        "displayName-count-other": "Trinidad und Tobago-Dollar",
        "symbol": "TTD",
        "symbol-alt-narrow": "$"
    },
    "TWD": {
        "displayName": "Neuer Taiwan-Dollar",
        "displayName-count-one": "Neuer Taiwan-Dollar",
        "displayName-count-other": "Neue Taiwan-Dollar",
        "symbol": "NT$",
        "symbol-alt-narrow": "NT$"
    },
    "TZS": {
        "displayName": "Tansania-Schilling",
        "displayName-count-one": "Tansania-Schilling",
        "displayName-count-other": "Tansania-Schilling",
        "symbol": "TZS"
    },
    "UAH": {
        "displayName": "Ukrainische Hrywnja",
        "displayName-count-one": "Ukrainische Hrywnja",
        "displayName-count-other": "Ukrainische Hrywen",
        "symbol": "UAH",
        "symbol-alt-narrow": "₴"
    },
    "UAK": {
        "displayName": "Ukrainischer Karbovanetz",
        "displayName-count-one": "Ukrainische Karbovanetz",
        "displayName-count-other": "Ukrainische Karbovanetz",
        "symbol": "UAK"
    }

```

```

    },
    "UGS": {
      "displayName": "Uganda-Schilling (1966-1987)",
      "displayName-count-one": "Uganda-Schilling (1966-1987)",
      "displayName-count-other": "Uganda-Schilling (1966-1987)",
      "symbol": "UGS"
    },
    "UGX": {
      "displayName": "Uganda-Schilling",
      "displayName-count-one": "Uganda-Schilling",
      "displayName-count-other": "Uganda-Schilling",
      "symbol": "UGX"
    },
    "USD": {
      "displayName": "US-Dollar",
      "displayName-count-one": "US-Dollar",
      "displayName-count-other": "US-Dollar",
      "symbol": "$",
      "symbol-alt-narrow": "$"
    },
    "USN": {
      "displayName": "US Dollar (Nächster Tag)",
      "displayName-count-one": "US-Dollar (Nächster Tag)",
      "displayName-count-other": "US-Dollar (Nächster Tag)",
      "symbol": "USN"
    },
    "USS": {
      "displayName": "US Dollar (Gleicher Tag)",
      "displayName-count-one": "US-Dollar (Gleicher Tag)",
      "displayName-count-other": "US-Dollar (Gleicher Tag)",
      "symbol": "USS"
    },
    "UYI": {
      "displayName": "Uruguayischer Peso (Indexierte
Rechnungseinheiten)",
      "displayName-count-one": "Uruguayischer Peso (Indexierte
Rechnungseinheiten)",
      "displayName-count-other": "Uruguayische Pesos (Indexierte
Rechnungseinheiten)",
      "symbol": "UYI"
    },
    "UYP": {
      "displayName": "Uruguayischer Peso (1975-1993)",
      "displayName-count-one": "Uruguayischer Peso (1975-1993)",
      "displayName-count-other": "Uruguayische Pesos (1975-1993)",
      "symbol": "UYP"
    },
    "UYU": {
      "displayName": "Uruguayischer Peso",
      "displayName-count-one": "Uruguayischer Peso",
      "displayName-count-other": "Uruguayische Pesos",
      "symbol": "UYU",
      "symbol-alt-narrow": "$"
    },
    "UZS": {
      "displayName": "Usbekistan-Sum",
      "displayName-count-one": "Usbekistan-Sum",

```

```

        "displayName-count-other": "Usbekistan-Sum",
        "symbol": "UZS"
    },
    "VEB": {
        "displayName": "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-one": "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-other": "Venezolanische Bolívares (1871-
2008)",
        "symbol": "VEB"
    },
    "VEF": {
        "displayName": "Venezolanischer Bolívar",
        "displayName-count-one": "Venezolanischer Bolívar",
        "displayName-count-other": "Venezolanische Bolívares",
        "symbol": "VEF",
        "symbol-alt-narrow": "Bs"
    },
    "VND": {
        "displayName": "Vietnamesischer Dong",
        "displayName-count-one": "Vietnamesischer Dong",
        "displayName-count-other": "Vietnamesische Dong",
        "symbol": "₫",
        "symbol-alt-narrow": "₫"
    },
    "VNN": {
        "displayName": "Vietnamesischer Dong(1978-1985)",
        "displayName-count-one": "Vietnamesischer Dong(1978-1985)",
        "displayName-count-other": "Vietnamesische Dong(1978-1985)",
        "symbol": "VNN"
    },
    "VUV": {
        "displayName": "Vanuatu-Vatu",
        "displayName-count-one": "Vanuatu-Vatu",
        "displayName-count-other": "Vanuatu-Vatu",
        "symbol": "VUV"
    },
    "WST": {
        "displayName": "Samoanischer Tala",
        "displayName-count-one": "Samoanischer Tala",
        "displayName-count-other": "Samoanische Tala",
        "symbol": "WST"
    },
    "XAF": {
        "displayName": "CFA-Franc (BEAC)",
        "displayName-count-one": "CFA-Franc (BEAC)",
        "displayName-count-other": "CFA-Franc (BEAC)",
        "symbol": "FCFA"
    },
    "XAG": {
        "displayName": "Unze Silber",
        "displayName-count-one": "Unze Silber",
        "displayName-count-other": "Unzen Silber",
        "symbol": "XAG"
    },
    "XAU": {
        "displayName": "Unze Gold",
        "displayName-count-one": "Unze Gold",

```

```

        "displayName-count-other": "Unzen Gold",
        "symbol": "XAU"
    },
    "XBA": {
        "displayName": "Europäische Rechnungseinheit",
        "displayName-count-one": "Europäische Rechnungseinheiten",
        "displayName-count-other": "Europäische Rechnungseinheiten",
        "symbol": "XBA"
    },
    "XBB": {
        "displayName": "Europäische Währungseinheit (XBB)",
        "displayName-count-one": "Europäische Währungseinheiten (XBB)",
        "displayName-count-other": "Europäische Währungseinheiten (XBB)",
        "symbol": "XBB"
    },
    "XBC": {
        "displayName": "Europäische Rechnungseinheit (XBC)",
        "displayName-count-one": "Europäische Rechnungseinheiten (XBC)",
        "displayName-count-other": "Europäische Rechnungseinheiten
(XBC) ",
        "symbol": "XBC"
    },
    "XBD": {
        "displayName": "Europäische Rechnungseinheit (XBD)",
        "displayName-count-one": "Europäische Rechnungseinheiten (XBD)",
        "displayName-count-other": "Europäische Rechnungseinheiten
(XBD) ",
        "symbol": "XBD"
    },
    "XCD": {
        "displayName": "Ostkaribischer Dollar",
        "displayName-count-one": "Ostkaribischer Dollar",
        "displayName-count-other": "Ostkaribische Dollar",
        "symbol": "EC$",
        "symbol-alt-narrow": "$"
    },
    "XDR": {
        "displayName": "Sonderziehungsrechte",
        "displayName-count-one": "Sonderziehungsrechte",
        "displayName-count-other": "Sonderziehungsrechte",
        "symbol": "XDR"
    },
    "XEU": {
        "displayName": "Europäische Währungseinheit (XEU)",
        "displayName-count-one": "Europäische Währungseinheiten (XEU)",
        "displayName-count-other": "Europäische Währungseinheiten (XEU)",
        "symbol": "XEU"
    },
    "XFO": {
        "displayName": "Französischer Gold-Franc",
        "displayName-count-one": "Französische Gold-Franc",
        "displayName-count-other": "Französische Gold-Franc",
        "symbol": "XFO"
    },
    "XFU": {
        "displayName": "Französischer UIC-Franc",
        "displayName-count-one": "Französische UIC-Franc",

```

```

    "displayName-count-other": "Französische UIC-Franc",
    "symbol": "XFU"
  },
  "XOF": {
    "displayName": "CFA-Franc (BCEAO)",
    "displayName-count-one": "CFA-Franc (BCEAO)",
    "displayName-count-other": "CFA-Francs (BCEAO)",
    "symbol": "CFA"
  },
  "XPD": {
    "displayName": "Unze Palladium",
    "displayName-count-one": "Unze Palladium",
    "displayName-count-other": "Unzen Palladium",
    "symbol": "XPD"
  },
  "XPF": {
    "displayName": "CFP-Franc",
    "displayName-count-one": "CFP-Franc",
    "displayName-count-other": "CFP-Franc",
    "symbol": "CFPF"
  },
  "XPT": {
    "displayName": "Unze Platin",
    "displayName-count-one": "Unze Platin",
    "displayName-count-other": "Unzen Platin",
    "symbol": "XPT"
  },
  "XRE": {
    "displayName": "RINET Funds",
    "displayName-count-one": "RINET Funds",
    "displayName-count-other": "RINET Funds",
    "symbol": "XRE"
  },
  "XSU": {
    "displayName": "SUCRE",
    "displayName-count-one": "SUCRE",
    "displayName-count-other": "SUCRE",
    "symbol": "XSU"
  },
  "XTS": {
    "displayName": "Testwährung",
    "displayName-count-one": "Testwährung",
    "displayName-count-other": "Testwährung",
    "symbol": "XTS"
  },
  "XUA": {
    "displayName": "Rechnungseinheit der AfEB",
    "displayName-count-one": "Rechnungseinheit der AfEB",
    "displayName-count-other": "Rechnungseinheiten der AfEB",
    "symbol": "XUA"
  },
  "XXX": {
    "displayName": "Unbekannte Währung",
    "displayName-count-one": "(unbekannte Währung)",
    "displayName-count-other": "(unbekannte Währung)",
    "symbol": "XXX"
  },
},

```



```

    "YDD": {
      "displayName": "Jemen-Dinar",
      "displayName-count-one": "Jemen-Dinar",
      "displayName-count-other": "Jemen-Dinar",
      "symbol": "YDD"
    },
    "YER": {
      "displayName": "Jemen-Rial",
      "displayName-count-one": "Jemen-Rial",
      "displayName-count-other": "Jemen-Rial",
      "symbol": "YER"
    },
    "YUD": {
      "displayName": "Jugoslawischer Dinar (1966-1990)",
      "displayName-count-one": "Jugoslawischer Dinar (1966-1990)",
      "displayName-count-other": "Jugoslawische Dinar (1966-1990)",
      "symbol": "YUD"
    },
    "YUM": {
      "displayName": "Jugoslawischer Neuer Dinar (1994-2002)",
      "displayName-count-one": "Jugoslawischer Neuer Dinar (1994-
2002)",
      "displayName-count-other": "Jugoslawische Neue Dinar (1994-
2002)",
      "symbol": "YUM"
    },
    "YUN": {
      "displayName": "Jugoslawischer Dinar (konvertibel)",
      "displayName-count-one": "Jugoslawische Dinar (konvertibel)",
      "displayName-count-other": "Jugoslawische Dinar (konvertibel)",
      "symbol": "YUN"
    },
    "YUR": {
      "displayName": "Jugoslawischer reformierter Dinar (1992-1993)",
      "displayName-count-one": "Jugoslawischer reformierter Dinar
(1992-1993)",
      "displayName-count-other": "Jugoslawische reformierte Dinar
(1992-1993)",
      "symbol": "YUR"
    },
    "ZAL": {
      "displayName": "Südafrikanischer Rand (Finanz)",
      "displayName-count-one": "Südafrikanischer Rand (Finanz)",
      "displayName-count-other": "Südafrikanischer Rand (Finanz)",
      "symbol": "ZAL"
    },
    "ZAR": {
      "displayName": "Südafrikanischer Rand",
      "displayName-count-one": "Südafrikanischer Rand",
      "displayName-count-other": "Südafrikanische Rand",
      "symbol": "ZAR",
      "symbol-alt-narrow": "R"
    },
    "ZMK": {
      "displayName": "Kwacha (1968-2012)",
      "displayName-count-one": "Kwacha (1968-2012)",
      "displayName-count-other": "Kwacha (1968-2012)",

```

```

    "symbol": "ZMK"
  },
  "ZMW": {
    "displayName": "Kwacha",
    "displayName-count-one": "Kwacha",
    "displayName-count-other": "Kwacha",
    "symbol": "ZMW",
    "symbol-alt-narrow": "K"
  },
  "ZRN": {
    "displayName": "Zaire-Neuer Zaïre (1993-1998)",
    "displayName-count-one": "Zaire-Neuer Zaïre (1993-1998)",
    "displayName-count-other": "Zaire-Neue Zaïre (1993-1998)",
    "symbol": "ZRN"
  },
  "ZRZ": {
    "displayName": "Zaire-Zaïre (1971-1993)",
    "displayName-count-one": "Zaire-Zaïre (1971-1993)",
    "displayName-count-other": "Zaire-Zaïre (1971-1993)",
    "symbol": "ZRZ"
  },
  "ZWD": {
    "displayName": "Simbabwe-Dollar (1980-2008)",
    "displayName-count-one": "Simbabwe-Dollar (1980-2008)",
    "displayName-count-other": "Simbabwe-Dollar (1980-2008)",
    "symbol": "ZWD"
  },
  "ZWL": {
    "displayName": "Simbabwe-Dollar (2009)",
    "displayName-count-one": "Simbabwe-Dollar (2009)",
    "displayName-count-other": "Simbabwe-Dollar (2009)",
    "symbol": "ZWL"
  },
  "ZWR": {
    "displayName": "Simbabwe-Dollar (2008)",
    "displayName-count-one": "Simbabwe-Dollar (2008)",
    "displayName-count-other": "Simbabwe-Dollar (2008)",
    "symbol": "ZWR"
  }
}

```

CURRENCIES.JSX

```
{
  "main":
  {
    "de":
    {
      "identity":
      {
        "version":
        {
```

```

        "_number";
        "$Revision: 13259 $",
        "_cldrVersion";
        "31";
    }
    "language";
    "de";
}
"numbers";
{
    "currencies";
    {
        "ADP";
        {
            "displayName";
            "Andorranische Pesete",
            "displayName-count-one";
            "Andorranische Pesete",
            "displayName-count-other";
            "Andorranische Peseten",
            "symbol";
            "ADP";
        }
        "AED";
        {
            "displayName";
            "VAE-Dirham",
            "displayName-count-one";
            "VAE-Dirham",
            "displayName-count-other";
            "VAE-Dirham",
            "symbol";
            "AED";
        }
        "AFA";
        {
            "displayName";
            "Afghanische Afghani (1927-2002)",
            "displayName-count-one";
            "Afghanische Afghani (1927-2002)",
            "displayName-count-other";
            "Afghanische Afghani (1927-2002)",
            "symbol";
            "AFA";
        }
        "AFN";
        {
            "displayName";
            "Afghanischer Afghani",
            "displayName-count-one";
            "Afghanischer Afghani",
            "displayName-count-other";
            "Afghanische Afghani",
            "symbol";
            "AFN";
        }
        "ALK";
    }
}

```

```

    {
      "displayName";
      "Albanischer Lek (1946-1965)",
      "displayName-count-one";
      "Albanischer Lek (1946-1965)",
      "displayName-count-other";
      "Albanische Lek (1946-1965)";
    }
    "ALL";
    {
      "displayName";
      "Albanischer Lek",
      "displayName-count-one";
      "Albanischer Lek",
      "displayName-count-other";
      "Albanische Lek",
      "symbol";
      "ALL";
    }
    "AMD";
    {
      "displayName";
      "Armenischer Dram",
      "displayName-count-one";
      "Armenischer Dram",
      "displayName-count-other";
      "Armenische Dram",
      "symbol";
      "AMD";
    }
    "ANG";
    {
      "displayName";
      "Niederländische-Antillen-Gulden",
      "displayName-count-one";
      "Niederländische-Antillen-Gulden",
      "displayName-count-other";
      "Niederländische-Antillen-Gulden",
      "symbol";
      "ANG";
    }
    "AOA";
    {
      "displayName";
      "Angolanischer Kwanza",
      "displayName-count-one";
      "Angolanischer Kwanza",
      "displayName-count-other";
      "Angolanische Kwanza",
      "symbol";
      "AOA",
      "symbol-alt-narrow";
      "Kz";
    }
    "AOK";
    {
      "displayName";

```

```

        "Angolanischer Kwanza (1977-1990)",
        "displayName-count-one";
        "Angolanischer Kwanza (1977-1990)",
        "displayName-count-other";
        "Angolanische Kwanza (1977-1990)",
        "symbol";
        "AOK";
    }
    "AON";
    {
        "displayName";
        "Angolanischer Neuer Kwanza (1990-2000)",
        "displayName-count-one";
        "Angolanischer Neuer Kwanza (1990-2000)",
        "displayName-count-other";
        "Angolanische Neue Kwanza (1990-2000)",
        "symbol";
        "AON";
    }
    "AOR";
    {
        "displayName";
        "Angolanischer Kwanza Reajustado (1995-1999)",
        "displayName-count-one";
        "Angolanischer Kwanza Reajustado (1995-1999)",
        "displayName-count-other";
        "Angolanische Kwanza Reajustado (1995-1999)",
        "symbol";
        "AOR";
    }
    "ARA";
    {
        "displayName";
        "Argentinischer Austral",
        "displayName-count-one";
        "Argentinischer Austral",
        "displayName-count-other";
        "Argentinische Austral",
        "symbol";
        "ARA";
    }
    "ARL";
    {
        "displayName";
        "Argentinischer Peso Ley (1970-1983)",
        "displayName-count-one";
        "Argentinischer Peso Ley (1970-1983)",
        "displayName-count-other";
        "Argentinische Pesos Ley (1970-1983)",
        "symbol";
        "ARL";
    }
    "ARM";
    {
        "displayName";
        "Argentinischer Peso (1881-1970)",
        "displayName-count-one";

```

```

        "Argentinischer Peso (1881-1970)",
        "displayName-count-other";
        "Argentinische Pesos (1881-1970)",
        "symbol";
        "ARM";
    }
    "ARP";
    {
        "displayName";
        "Argentinischer Peso (1983-1985)",
        "displayName-count-one";
        "Argentinischer Peso (1983-1985)",
        "displayName-count-other";
        "Argentinische Peso (1983-1985)",
        "symbol";
        "ARP";
    }
    "ARS";
    {
        "displayName";
        "Argentinischer Peso",
        "displayName-count-one";
        "Argentinischer Peso",
        "displayName-count-other";
        "Argentinische Pesos",
        "symbol";
        "ARS",
        "symbol-alt-narrow";
        "$";
    }
    "ATS";
    {
        "displayName";
        "Österreichischer Schilling",
        "displayName-count-one";
        "Österreichischer Schilling",
        "displayName-count-other";
        "Österreichische Schilling",
        "symbol";
        "öS";
    }
    "AUD";
    {
        "displayName";
        "Australischer Dollar",
        "displayName-count-one";
        "Australischer Dollar",
        "displayName-count-other";
        "Australische Dollar",
        "symbol";
        "AU$",
        "symbol-alt-narrow";
        "$";
    }
    "AWG";
    {
        "displayName";

```

```

        "Aruba-Florin",
        "displayName-count-one";
        "Aruba-Florin",
        "displayName-count-other";
        "Aruba-Florin",
        "symbol";
        "AWG";
    }
    "AZM";
    {
        "displayName";
        "Aserbajdschan-Manat (1993-2006)",
        "displayName-count-one";
        "Aserbajdschan-Manat (1993-2006)",
        "displayName-count-other";
        "Aserbajdschan-Manat (1993-2006)",
        "symbol";
        "AZM";
    }
    "AZN";
    {
        "displayName";
        "Aserbajdschan-Manat",
        "displayName-count-one";
        "Aserbajdschan-Manat",
        "displayName-count-other";
        "Aserbajdschan-Manat",
        "symbol";
        "AZN";
    }
    "BAD";
    {
        "displayName";
        "Bosnien und Herzegowina Dinar (1992-1994)",
        "displayName-count-one";
        "Bosnien und Herzegowina Dinar (1992-1994)",
        "displayName-count-other";
        "Bosnien und Herzegowina Dinar (1992-1994)",
        "symbol";
        "BAD";
    }
    "BAM";
    {
        "displayName";
        "Bosnien und Herzegowina Konvertierbare Mark",
        "displayName-count-one";
        "Bosnien und Herzegowina Konvertierbare Mark",
        "displayName-count-other";
        "Bosnien und Herzegowina Konvertierbare Mark",
        "symbol";
        "BAM",
        "symbol-alt-narrow";
        "KM";
    }
    "BAN";
    {
        "displayName";

```

```

        "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
        "displayName-count-one";
        "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
        "displayName-count-other";
        "Bosnien und Herzegowina Neue Dinar (1994-1997)",
        "symbol";
        "BAN";
    }
    "BBD";
    {
        "displayName";
        "Barbados-Dollar",
        "displayName-count-one";
        "Barbados-Dollar",
        "displayName-count-other";
        "Barbados-Dollar",
        "symbol";
        "BBD",
        "symbol-alt-narrow";
        "$";
    }
    "BDT";
    {
        "displayName";
        "Bangladesch-Taka",
        "displayName-count-one";
        "Bangladesch-Taka",
        "displayName-count-other";
        "Bangladesch-Taka",
        "symbol";
        "BDT",
        "symbol-alt-narrow";
        "৳";
    }
    "BEC";
    {
        "displayName";
        "Belgischer Franc (konvertibel)",
        "displayName-count-one";
        "Belgischer Franc (konvertibel)",
        "displayName-count-other";
        "Belgische Franc (konvertibel)",
        "symbol";
        "BEC";
    }
    "BEF";
    {
        "displayName";
        "Belgischer Franc",
        "displayName-count-one";
        "Belgischer Franc",
        "displayName-count-other";
        "Belgische Franc",
        "symbol";
        "BEF";
    }
}

```



```
"BEL";
{
  "displayName";
  "Belgischer Finanz-Franc",
  "displayName-count-one";
  "Belgischer Finanz-Franc",
  "displayName-count-other";
  "Belgische Finanz-Franc",
  "symbol";
  "BEL";
}
"BGL";
{
  "displayName";
  "Bulgarische Lew (1962-1999)",
  "displayName-count-one";
  "Bulgarische Lew (1962-1999)",
  "displayName-count-other";
  "Bulgarische Lew (1962-1999)",
  "symbol";
  "BGL";
}
"BGM";
{
  "displayName";
  "Bulgarischer Lew (1952-1962)",
  "displayName-count-one";
  "Bulgarischer Lew (1952-1962)",
  "displayName-count-other";
  "Bulgarische Lew (1952-1962)",
  "symbol";
  "BGK";
}
"BGN";
{
  "displayName";
  "Bulgarischer Lew",
  "displayName-count-one";
  "Bulgarischer Lew",
  "displayName-count-other";
  "Bulgarische Lew",
  "symbol";
  "BGN";
}
"BGO";
{
  "displayName";
  "Bulgarischer Lew (1879-1952)",
  "displayName-count-one";
  "Bulgarischer Lew (1879-1952)",
  "displayName-count-other";
  "Bulgarische Lew (1879-1952)",
  "symbol";
  "BGJ";
}
"BHD";
{
```

```

        "displayName";
        "Bahrain-Dinar",
        "displayName-count-one";
        "Bahrain-Dinar",
        "displayName-count-other";
        "Bahrain-Dinar",
        "symbol";
        "BHD";
    }
    "BIF";
    {
        "displayName";
        "Burundi-Franc",
        "displayName-count-one";
        "Burundi-Franc",
        "displayName-count-other";
        "Burundi-Francis",
        "symbol";
        "BIF";
    }
    "BMD";
    {
        "displayName";
        "Bermuda-Dollar",
        "displayName-count-one";
        "Bermuda-Dollar",
        "displayName-count-other";
        "Bermuda-Dollar",
        "symbol";
        "BMD",
        "symbol-alt-narrow";
        "$";
    }
    "BND";
    {
        "displayName";
        "Brunei-Dollar",
        "displayName-count-one";
        "Brunei-Dollar",
        "displayName-count-other";
        "Brunei-Dollar",
        "symbol";
        "BND",
        "symbol-alt-narrow";
        "$";
    }
    "BOB";
    {
        "displayName";
        "Bolivanischer Boliviano",
        "displayName-count-one";
        "Bolivanischer Boliviano",
        "displayName-count-other";
        "Bolivianische Bolivianos",
        "symbol";
        "BOB",
        "symbol-alt-narrow";
    }

```

```

        "Bs";
    }
    "BOL";
    {
        "displayName";
        "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-one";
        "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-other";
        "Bolivianische Bolivianos (1863-1963)",
        "symbol";
        "BOL";
    }
    "BOP";
    {
        "displayName";
        "Bolivianischer Peso",
        "displayName-count-one";
        "Bolivianischer Peso",
        "displayName-count-other";
        "Bolivianische Peso",
        "symbol";
        "BOP";
    }
    "BOV";
    {
        "displayName";
        "Boliviensische Mvdol",
        "displayName-count-one";
        "Boliviensische Mvdol",
        "displayName-count-other";
        "Bolivianische Mvdol",
        "symbol";
        "BOV";
    }
    "BRB";
    {
        "displayName";
        "Brasilianischer Cruzeiro Novo (1967-1986)",
        "displayName-count-one";
        "Brasilianischer Cruzeiro Novo (1967-1986)",
        "displayName-count-other";
        "Brasilianische Cruzeiro Novo (1967-1986)",
        "symbol";
        "BRB";
    }
    "BRC";
    {
        "displayName";
        "Brasilianischer Cruzado (1986-1989)",
        "displayName-count-one";
        "Brasilianischer Cruzado (1986-1989)",
        "displayName-count-other";
        "Brasilianische Cruzado (1986-1989)",
        "symbol";
        "BRC";
    }
}

```

```
"BRE";
{
  "displayName";
  "Brasilianischer Cruzeiro (1990-1993)",
    "displayName-count-one";
  "Brasilianischer Cruzeiro (1990-1993)",
    "displayName-count-other";
  "Brasilianische Cruzeiro (1990-1993)",
    "symbol";
  "BRE";
}
"BRL";
{
  "displayName";
  "Brasilianischer Real",
    "displayName-count-one";
  "Brasilianischer Real",
    "displayName-count-other";
  "Brasilianische Real",
    "symbol";
  "R$",
    "symbol-alt-narrow";
  "R$";
}
"BRN";
{
  "displayName";
  "Brasilianischer Cruzado Novo (1989-1990)",
    "displayName-count-one";
  "Brasilianischer Cruzado Novo (1989-1990)",
    "displayName-count-other";
  "Brasilianische Cruzado Novo (1989-1990)",
    "symbol";
  "BRN";
}
"BRR";
{
  "displayName";
  "Brasilianischer Cruzeiro (1993-1994)",
    "displayName-count-one";
  "Brasilianischer Cruzeiro (1993-1994)",
    "displayName-count-other";
  "Brasilianische Cruzeiro (1993-1994)",
    "symbol";
  "BRR";
}
"BRZ";
{
  "displayName";
  "Brasilianischer Cruzeiro (1942-1967)",
    "displayName-count-one";
  "Brasilianischer Cruzeiro (1942-1967)",
    "displayName-count-other";
  "Brasilianischer Cruzeiro (1942-1967)",
    "symbol";
  "BRZ";
}
```

```

"BSD";
{
  "displayName";
  "Bahamas-Dollar",
    "displayName-count-one";
  "Bahamas-Dollar",
    "displayName-count-other";
  "Bahamas-Dollar",
    "symbol";
  "BSD",
    "symbol-alt-narrow";
  "$";
}
"BTN";
{
  "displayName";
  "Bhutan-Ngultrum",
    "displayName-count-one";
  "Bhutan-Ngultrum",
    "displayName-count-other";
  "Bhutan-Ngultrum",
    "symbol";
  "BTN";
}
"BUK";
{
  "displayName";
  "Birmanischer Kyat",
    "displayName-count-one";
  "Birmanischer Kyat",
    "displayName-count-other";
  "Birmanische Kyat",
    "symbol";
  "BUK";
}
"BWP";
{
  "displayName";
  "Botswanischer Pula",
    "displayName-count-one";
  "Botswanischer Pula",
    "displayName-count-other";
  "Botswanische Pula",
    "symbol";
  "BWP",
    "symbol-alt-narrow";
  "P";
}
"BYB";
{
  "displayName";
  "Belarus-Rubel (1994-1999)",
    "displayName-count-one";
  "Belarus-Rubel (1994-1999)",
    "displayName-count-other";
  "Belarus-Rubel (1994-1999)",
    "symbol";
}

```

```

        "BYB";
    }
    "BYN";
    {
        "displayName";
        "Weißrussischer Rubel",
        "displayName-count-one";
        "Weißrussischer Rubel",
        "displayName-count-other";
        "Weißrussische Rubel",
        "symbol";
        "BYN",
        "symbol-alt-narrow";
        "p.";
    }
    "BYR";
    {
        "displayName";
        "Weißrussischer Rubel (2000-2016)",
        "displayName-count-one";
        "Weißrussischer Rubel (2000-2016)",
        "displayName-count-other";
        "Weißrussische Rubel (2000-2016)",
        "symbol";
        "BYR";
    }
    "BZD";
    {
        "displayName";
        "Belize-Dollar",
        "displayName-count-one";
        "Belize-Dollar",
        "displayName-count-other";
        "Belize-Dollar",
        "symbol";
        "BZD",
        "symbol-alt-narrow";
        "$";
    }
    "CAD";
    {
        "displayName";
        "Kanadischer Dollar",
        "displayName-count-one";
        "Kanadischer Dollar",
        "displayName-count-other";
        "Kanadische Dollar",
        "symbol";
        "CA$",
        "symbol-alt-narrow";
        "$";
    }
    "CDF";
    {
        "displayName";
        "Kongo-Franc",
        "displayName-count-one";

```

```

        "Kongo-Franc",
        "displayName-count-other";
        "Kongo-Francs",
        "symbol";
        "CDF";
    }
    "CHE";
    {
        "displayName";
        "WIR-Euro",
        "displayName-count-one";
        "WIR-Euro",
        "displayName-count-other";
        "WIR-Euro",
        "symbol";
        "CHE";
    }
    "CHF";
    {
        "displayName";
        "Schweizer Franken",
        "displayName-count-one";
        "Schweizer Franken",
        "displayName-count-other";
        "Schweizer Franken",
        "symbol";
        "CHF";
    }
    "CHW";
    {
        "displayName";
        "WIR Franken",
        "displayName-count-one";
        "WIR Franken",
        "displayName-count-other";
        "WIR Franken",
        "symbol";
        "CHW";
    }
    "CLE";
    {
        "displayName";
        "Chilenischer Escudo",
        "displayName-count-one";
        "Chilenischer Escudo",
        "displayName-count-other";
        "Chilenische Escudo",
        "symbol";
        "CLE";
    }
    "CLF";
    {
        "displayName";
        "Chilenische Unidades de Fomento",
        "displayName-count-one";
        "Chilenische Unidades de Fomento",
        "displayName-count-other";
    }

```

```

        "Chilenische Unidades de Fomento",
        "symbol";
        "CLF";
    }
    "CLP";
    {
        "displayName";
        "Chilenischer Peso",
        "displayName-count-one";
        "Chilenischer Peso",
        "displayName-count-other";
        "Chilenische Pesos",
        "symbol";
        "CLP",
        "symbol-alt-narrow";
        "$";
    }
    "CNX";
    {
        "displayName";
        "Dollar der Chinesischen Volksbank",
        "displayName-count-one";
        "Dollar der Chinesischen Volksbank",
        "displayName-count-other";
        "Dollar der Chinesischen Volksbank",
        "symbol";
        "CNX";
    }
    "CNY";
    {
        "displayName";
        "Renminbi Yuan",
        "displayName-count-one";
        "Chinesischer Yuan",
        "displayName-count-other";
        "Renminbi Yuan",
        "symbol";
        "CN¥",
        "symbol-alt-narrow";
        "¥";
    }
    "COP";
    {
        "displayName";
        "Kolumbianischer Peso",
        "displayName-count-one";
        "Kolumbianischer Peso",
        "displayName-count-other";
        "Kolumbianische Pesos",
        "symbol";
        "COP",
        "symbol-alt-narrow";
        "$";
    }
    "COU";
    {
        "displayName";

```



```

        "Kolumbianische Unidades de valor real",
        "displayName-count-one";
        "Kolumbianische Unidad de valor real",
        "displayName-count-other";
        "Kolumbianische Unidades de valor real",
        "symbol";
        "COU";
    }
    "CRC";
    {
        "displayName";
        "Costa-Rica-Colón",
        "displayName-count-one";
        "Costa-Rica-Colón",
        "displayName-count-other";
        "Costa-Rica-Colón",
        "symbol";
        "CRC",
        "symbol-alt-narrow";
        "₡";
    }
    "CSD";
    {
        "displayName";
        "Serbischer Dinar (2002-2006)",
        "displayName-count-one";
        "Serbischer Dinar (2002-2006)",
        "displayName-count-other";
        "Serbische Dinar (2002-2006)",
        "symbol";
        "CSD";
    }
    "CSK";
    {
        "displayName";
        "Tschechoslowakische Krone",
        "displayName-count-one";
        "Tschechoslowakische Kronen",
        "displayName-count-other";
        "Tschechoslowakische Kronen",
        "symbol";
        "CSK";
    }
    "CUC";
    {
        "displayName";
        "Kubanischer Peso (konvertibel)",
        "displayName-count-one";
        "Kubanischer Peso (konvertibel)",
        "displayName-count-other";
        "Kubanische Pesos (konvertibel)",
        "symbol";
        "CUC",
        "symbol-alt-narrow";
        "Cub$";
    }
    "CUP";

```

```
{
  "displayName";
  "Kubanischer Peso",
    "displayName-count-one";
  "Kubanischer Peso",
    "displayName-count-other";
  "Kubanische Pesos",
    "symbol";
  "CUP",
    "symbol-alt-narrow";
  "$";
}
"CVE";
{
  "displayName";
  "Cabo-Verde-Escudo",
    "displayName-count-one";
  "Cabo-Verde-Escudo",
    "displayName-count-other";
  "Cabo-Verde-Escudos",
    "symbol";
  "CVE";
}
"CYP";
{
  "displayName";
  "Zypern-Pfund",
    "displayName-count-one";
  "Zypern Pfund",
    "displayName-count-other";
  "Zypern Pfund",
    "symbol";
  "CYP";
}
"CZK";
{
  "displayName";
  "Tschechische Krone",
    "displayName-count-one";
  "Tschechische Krone",
    "displayName-count-other";
  "Tschechische Kronen",
    "symbol";
  "CZK",
    "symbol-alt-narrow";
  "Kč";
}
"DDM";
{
  "displayName";
  "Mark der DDR",
    "displayName-count-one";
  "Mark der DDR",
    "displayName-count-other";
  "Mark der DDR",
    "symbol";
  "DDM";
}
```

```

    }
    "DEM";
    {
        "displayName";
        "Deutsche Mark",
            "displayName-count-one";
        "Deutsche Mark",
            "displayName-count-other";
        "Deutsche Mark",
            "symbol";
        "DM";
    }
    "DJF";
    {
        "displayName";
        "Dschibuti-Franc",
            "displayName-count-one";
        "Dschibuti-Franc",
            "displayName-count-other";
        "Dschibuti-Franc",
            "symbol";
        "DJF";
    }
    "DKK";
    {
        "displayName";
        "Dänische Krone",
            "displayName-count-one";
        "Dänische Krone",
            "displayName-count-other";
        "Dänische Kronen",
            "symbol";
        "DKK",
            "symbol-alt-narrow";
        "kr";
    }
    "DOP";
    {
        "displayName";
        "Dominikanischer Peso",
            "displayName-count-one";
        "Dominikanischer Peso",
            "displayName-count-other";
        "Dominikanische Pesos",
            "symbol";
        "DOP",
            "symbol-alt-narrow";
        "$";
    }
    "DZD";
    {
        "displayName";
        "Algerischer Dinar",
            "displayName-count-one";
        "Algerischer Dinar",
            "displayName-count-other";
        "Algerische Dinar",

```

```

        "symbol";
        "DZD";
    }
    "ECS";
    {
        "displayName";
        "Ecuadorianischer Sucre",
        "displayName-count-one";
        "Ecuadorianischer Sucre",
        "displayName-count-other";
        "Ecuadorianische Sucre",
        "symbol";
        "ECS";
    }
    "ECV";
    {
        "displayName";
        "Verrechnungseinheit für Ecuador",
        "displayName-count-one";
        "Verrechnungseinheiten für Ecuador",
        "displayName-count-other";
        "Verrechnungseinheiten für Ecuador",
        "symbol";
        "ECV";
    }
    "EEK";
    {
        "displayName";
        "Estnische Krone",
        "displayName-count-one";
        "Estnische Krone",
        "displayName-count-other";
        "Estnische Kronen",
        "symbol";
        "EEK";
    }
    "EGP";
    {
        "displayName";
        "Ägyptisches Pfund",
        "displayName-count-one";
        "Ägyptisches Pfund",
        "displayName-count-other";
        "Ägyptische Pfund",
        "symbol";
        "EGP",
        "symbol-alt-narrow";
        "£";
    }
    "ERN";
    {
        "displayName";
        "Eritreischer Nakfa",
        "displayName-count-one";
        "Eritreischer Nakfa",
        "displayName-count-other";
        "Eritreische Nakfa",

```

```

        "symbol";
        "ERN";
    }
    "ESA";
    {
        "displayName";
        "Spanische Peseta (A-Konten)",
        "displayName-count-one";
        "Spanische Peseta (A-Konten)",
        "displayName-count-other";
        "Spanische Peseten (A-Konten)",
        "symbol";
        "ESA";
    }
    "ESB";
    {
        "displayName";
        "Spanische Peseta (konvertibel)",
        "displayName-count-one";
        "Spanische Peseta (konvertibel)",
        "displayName-count-other";
        "Spanische Peseten (konvertibel)",
        "symbol";
        "ESB";
    }
    "ESP";
    {
        "displayName";
        "Spanische Peseta",
        "displayName-count-one";
        "Spanische Peseta",
        "displayName-count-other";
        "Spanische Peseten",
        "symbol";
        "ESP",
        "symbol-alt-narrow";
        "₧";
    }
    "ETB";
    {
        "displayName";
        "Äthiopischer Birr",
        "displayName-count-one";
        "Äthiopischer Birr",
        "displayName-count-other";
        "Äthiopische Birr",
        "symbol";
        "ETB";
    }
    "EUR";
    {
        "displayName";
        "Euro",
        "displayName-count-one";
        "Euro",
        "displayName-count-other";
        "Euro",

```

```

        "symbol";
        "€",
        "symbol-alt-narrow";
        "€";
    }
    "FIM";
    {
        "displayName";
        "Finnische Mark",
        "displayName-count-one";
        "Finnische Mark",
        "displayName-count-other";
        "Finnische Mark",
        "symbol";
        "FIM";
    }
    "FJD";
    {
        "displayName";
        "Fidschi-Dollar",
        "displayName-count-one";
        "Fidschi-Dollar",
        "displayName-count-other";
        "Fidschi-Dollar",
        "symbol";
        "FJD",
        "symbol-alt-narrow";
        "$";
    }
    "FKP";
    {
        "displayName";
        "Falkland-Pfund",
        "displayName-count-one";
        "Falkland-Pfund",
        "displayName-count-other";
        "Falkland-Pfund",
        "symbol";
        "FKP",
        "symbol-alt-narrow";
        "Fl£";
    }
    "FRF";
    {
        "displayName";
        "Französischer Franc",
        "displayName-count-one";
        "Französischer Franc",
        "displayName-count-other";
        "Französische Franc",
        "symbol";
        "FRF";
    }
    "GBP";
    {
        "displayName";
        "Britisches Pfund",

```

```

        "displayName-count-one";
        "Britisches Pfund",
        "displayName-count-other";
        "Britische Pfund",
        "symbol";
        "£",
        "symbol-alt-narrow";
        "£";
    }
    "GEK";
    {
        "displayName";
        "Georgischer Kupon Larit",
        "displayName-count-one";
        "Georgischer Kupon Larit",
        "displayName-count-other";
        "Georgische Kupon Larit",
        "symbol";
        "GEK";
    }
    "GEL";
    {
        "displayName";
        "Georgischer Lari",
        "displayName-count-one";
        "Georgischer Lari",
        "displayName-count-other";
        "Georgische Lari",
        "symbol";
        "GEL",
        "symbol-alt-narrow";
        "ლ",
        "symbol-alt-variant";
        "ლ";
    }
    "GHC";
    {
        "displayName";
        "Ghanaischer Cedi (1979–2007)",
        "displayName-count-one";
        "Ghanaischer Cedi (1979–2007)",
        "displayName-count-other";
        "Ghanaische Cedi (1979–2007)",
        "symbol";
        "GHC";
    }
    "GHS";
    {
        "displayName";
        "Ghanaischer Cedi",
        "displayName-count-one";
        "Ghanaischer Cedi",
        "displayName-count-other";
        "Ghanaische Cedi",
        "symbol";
        "GHS";
    }
}

```

```

"GIP";
{
  "displayName";
  "Gibraltar-Pfund",
    "displayName-count-one";
  "Gibraltar-Pfund",
    "displayName-count-other";
  "Gibraltar Pfund",
    "symbol";
  "GIP",
    "symbol-alt-narrow";
  "£";
}
"GMD";
{
  "displayName";
  "Gambia-Dalasi",
    "displayName-count-one";
  "Gambia-Dalasi",
    "displayName-count-other";
  "Gambia-Dalasi",
    "symbol";
  "GMD";
}
"GNF";
{
  "displayName";
  "Guinea-Franc",
    "displayName-count-one";
  "Guinea-Franc",
    "displayName-count-other";
  "Guinea-Franc",
    "symbol";
  "GNF",
    "symbol-alt-narrow";
  "F.G.";
}
"GNS";
{
  "displayName";
  "Guineischer Syli",
    "displayName-count-one";
  "Guineischer Syli",
    "displayName-count-other";
  "Guineische Syli",
    "symbol";
  "GNS";
}
"GQE";
{
  "displayName";
  "Äquatorialguinea-Ekwele",
    "displayName-count-one";
  "Äquatorialguinea-Ekwele",
    "displayName-count-other";
  "Äquatorialguinea-Ekwele",
    "symbol";
}

```



```

        "GQE";
    }
    "GRD";
    {
        "displayName";
        "Griechische Drachme",
        "displayName-count-one";
        "Griechische Drachme",
        "displayName-count-other";
        "Griechische Drachmen",
        "symbol";
        "GRD";
    }
    "GTQ";
    {
        "displayName";
        "Guatemaltekinscher Quetzal",
        "displayName-count-one";
        "Guatemaltekinscher Quetzal",
        "displayName-count-other";
        "Guatemaltekinsche Quetzales",
        "symbol";
        "GTQ",
        "symbol-alt-narrow";
        "Q";
    }
    "GWE";
    {
        "displayName";
        "Portugiesisch Guinea Escudo",
        "displayName-count-one";
        "Portugiesisch Guinea Escudo",
        "displayName-count-other";
        "Portugiesisch Guinea Escudo",
        "symbol";
        "GWE";
    }
    "GWP";
    {
        "displayName";
        "Guinea-Bissau Peso",
        "displayName-count-one";
        "Guinea-Bissau Peso",
        "displayName-count-other";
        "Guinea-Bissau Pesos",
        "symbol";
        "GWP";
    }
    "GYD";
    {
        "displayName";
        "Guyana-Dollar",
        "displayName-count-one";
        "Guyana-Dollar",
        "displayName-count-other";
        "Guyana-Dollar",
        "symbol";
    }

```

```

        "GYD",
        "symbol-alt-narrow";
        "$";
    }
    "HKD";
    {
        "displayName";
        "Hongkong-Dollar",
        "displayName-count-one";
        "Hongkong-Dollar",
        "displayName-count-other";
        "Hongkong-Dollar",
        "symbol";
        "HK$";
        "symbol-alt-narrow";
        "$";
    }
    "HNL";
    {
        "displayName";
        "Honduras-Lempira",
        "displayName-count-one";
        "Honduras-Lempira",
        "displayName-count-other";
        "Honduras-Lempira",
        "symbol";
        "HNL",
        "symbol-alt-narrow";
        "L";
    }
    "HRD";
    {
        "displayName";
        "Kroatischer Dinar",
        "displayName-count-one";
        "Kroatischer Dinar",
        "displayName-count-other";
        "Kroatische Dinar",
        "symbol";
        "HRD";
    }
    "HRK";
    {
        "displayName";
        "Kroatischer Kuna",
        "displayName-count-one";
        "Kroatischer Kuna",
        "displayName-count-other";
        "Kroatische Kuna",
        "symbol";
        "HRK",
        "symbol-alt-narrow";
        "kn";
    }
    "HTG";
    {
        "displayName";

```

```

        "Haitianische Gourde",
        "displayName-count-one";
        "Haitianische Gourde",
        "displayName-count-other";
        "Haitianische Gourdes",
        "symbol";
        "HTG";
    }
    "HUF";
    {
        "displayName";
        "Ungarischer Forint",
        "displayName-count-one";
        "Ungarischer Forint",
        "displayName-count-other";
        "Ungarische Forint",
        "symbol";
        "HUF",
        "symbol-alt-narrow";
        "Ft";
    }
    "IDR";
    {
        "displayName";
        "Indonesische Rupiah",
        "displayName-count-one";
        "Indonesische Rupiah",
        "displayName-count-other";
        "Indonesische Rupiah",
        "symbol";
        "IDR",
        "symbol-alt-narrow";
        "Rp";
    }
    "IEP";
    {
        "displayName";
        "Irisches Pfund",
        "displayName-count-one";
        "Irisches Pfund",
        "displayName-count-other";
        "Irische Pfund",
        "symbol";
        "IEP";
    }
    "ILP";
    {
        "displayName";
        "Israelisches Pfund",
        "displayName-count-one";
        "Israelisches Pfund",
        "displayName-count-other";
        "Israelische Pfund",
        "symbol";
        "ILP";
    }
    "ILR";

```

```

    {
      "displayName";
      "Israelischer Schekel (1980-1985)",
      "displayName-count-one";
      "Israelischer Schekel (1980-1985)",
      "displayName-count-other";
      "Israelische Schekel (1980-1985)";
    }
    "ILS";
    {
      "displayName";
      "Israelischer Neuer Schekel",
      "displayName-count-one";
      "Israelischer Neuer Schekel",
      "displayName-count-other";
      "Israelische Neue Schekel",
      "symbol";
      "₪",
      "symbol-alt-narrow";
      "₪";
    }
    "INR";
    {
      "displayName";
      "Indische Rupie",
      "displayName-count-one";
      "Indische Rupie",
      "displayName-count-other";
      "Indische Rupien",
      "symbol";
      "₹",
      "symbol-alt-narrow";
      "₹";
    }
    "IQD";
    {
      "displayName";
      "Irakischer Dinar",
      "displayName-count-one";
      "Irakischer Dinar",
      "displayName-count-other";
      "Irakische Dinar",
      "symbol";
      "IQD";
    }
    "IRR";
    {
      "displayName";
      "Iranischer Rial",
      "displayName-count-one";
      "Iranischer Rial",
      "displayName-count-other";
      "Iranische Rial",
      "symbol";
      "IRR";
    }
    "ISJ";

```

```

    {
      "displayName";
      "Isländische Krone (1918-1981)",
      "displayName-count-one";
      "Isländische Krone (1918-1981)",
      "displayName-count-other";
      "Isländische Kronen (1918-1981)";
    }
    "ISK";
    {
      "displayName";
      "Isländische Krone",
      "displayName-count-one";
      "Isländische Krone",
      "displayName-count-other";
      "Isländische Kronen",
      "symbol";
      "ISK",
      "symbol-alt-narrow";
      "kr";
    }
    "ITL";
    {
      "displayName";
      "Italienische Lira",
      "displayName-count-one";
      "Italienische Lira",
      "displayName-count-other";
      "Italienische Lire",
      "symbol";
      "ITL";
    }
    "JMD";
    {
      "displayName";
      "Jamaika-Dollar",
      "displayName-count-one";
      "Jamaika-Dollar",
      "displayName-count-other";
      "Jamaika-Dollar",
      "symbol";
      "JMD",
      "symbol-alt-narrow";
      "$";
    }
    "JOD";
    {
      "displayName";
      "Jordanischer Dinar",
      "displayName-count-one";
      "Jordanischer Dinar",
      "displayName-count-other";
      "Jordanische Dinar",
      "symbol";
      "JOD";
    }
    "JPY";

```

```

    {
      "displayName";
      "Japanischer Yen",
        "displayName-count-one";
      "Japanischer Yen",
        "displayName-count-other";
      "Japanische Yen",
        "symbol";
      "¥",
        "symbol-alt-narrow";
      "¥";
    }
    "KES";
    {
      "displayName";
      "Kenia-Schilling",
        "displayName-count-one";
      "Kenia-Schilling",
        "displayName-count-other";
      "Kenia-Schilling",
        "symbol";
      "KES";
    }
    "KGS";
    {
      "displayName";
      "Kirgisischer Som",
        "displayName-count-one";
      "Kirgisischer Som",
        "displayName-count-other";
      "Kirgisische Som",
        "symbol";
      "KGS";
    }
    "KHR";
    {
      "displayName";
      "Kambodschanischer Riel",
        "displayName-count-one";
      "Kambodschanischer Riel",
        "displayName-count-other";
      "Kambodschanische Riel",
        "symbol";
      "KHR",
        "symbol-alt-narrow";
      "៛";
    }
    "KMF";
    {
      "displayName";
      "Komoren-Franc",
        "displayName-count-one";
      "Komoren-Franc",
        "displayName-count-other";
      "Komoren-Francs",
        "symbol";
    }

```

```

        "KMF",
        "symbol-alt-narrow";
        "FC";
    }
    "KPW";
    {
        "displayName";
        "Nordkoreanischer Won",
        "displayName-count-one";
        "Nordkoreanischer Won",
        "displayName-count-other";
        "Nordkoreanische Won",
        "symbol";
        "KPW",
        "symbol-alt-narrow";
        "₩";
    }
    "KRH";
    {
        "displayName";
        "Südkoreanischer Hwan (1953-1962)",
        "displayName-count-one";
        "Südkoreanischer Hwan (1953-1962)",
        "displayName-count-other";
        "Südkoreanischer Hwan (1953-1962)",
        "symbol";
        "KRH";
    }
    "KRO";
    {
        "displayName";
        "Südkoreanischer Won (1945-1953)",
        "displayName-count-one";
        "Südkoreanischer Won (1945-1953)",
        "displayName-count-other";
        "Südkoreanischer Won (1945-1953)",
        "symbol";
        "KRO";
    }
    "KRW";
    {
        "displayName";
        "Südkoreanischer Won",
        "displayName-count-one";
        "Südkoreanischer Won",
        "displayName-count-other";
        "Südkoreanische Won",
        "symbol";
        "₩",
        "symbol-alt-narrow";
        "₩";
    }
    "KWD";
    {
        "displayName";
        "Kuwait-Dinar",
        "displayName-count-one";

```

```

        "Kuwait-Dinar",
        "displayName-count-other";
    "Kuwait-Dinar",
        "symbol";
    "KWD";
}
"KYD";
{
    "displayName";
    "Kaiman-Dollar",
        "displayName-count-one";
    "Kaiman-Dollar",
        "displayName-count-other";
    "Kaiman-Dollar",
        "symbol";
    "KYD",
        "symbol-alt-narrow";
    "$";
}
"KZT";
{
    "displayName";
    "Kasachischer Tenge",
        "displayName-count-one";
    "Kasachischer Tenge",
        "displayName-count-other";
    "Kasachische Tenge",
        "symbol";
    "KZT",
        "symbol-alt-narrow";
    "Т";
}
"LAK";
{
    "displayName";
    "Laotischer Kip",
        "displayName-count-one";
    "Laotischer Kip",
        "displayName-count-other";
    "Laotische Kip",
        "symbol";
    "LAK",
        "symbol-alt-narrow";
    "₭";
}
"LBP";
{
    "displayName";
    "Libanesisches Pfund",
        "displayName-count-one";
    "Libanesisches Pfund",
        "displayName-count-other";
    "Libanesische Pfund",
        "symbol";
    "LBP",
        "symbol-alt-narrow";
    "₯";
}

```



```

    }
    "LKR";
    {
        "displayName";
        "Sri-Lanka-Rupie",
            "displayName-count-one";
        "Sri-Lanka-Rupie",
            "displayName-count-other";
        "Sri-Lanka-Rupien",
            "symbol";
        "LKR",
            "symbol-alt-narrow";
        "Rs";
    }
    "LRD";
    {
        "displayName";
        "Liberianischer Dollar",
            "displayName-count-one";
        "Liberianischer Dollar",
            "displayName-count-other";
        "Liberianische Dollar",
            "symbol";
        "LRD",
            "symbol-alt-narrow";
        "$";
    }
    "LSL";
    {
        "displayName";
        "Loti",
            "displayName-count-one";
        "Loti",
            "displayName-count-other";
        "Loti",
            "symbol";
        "LSL";
    }
    "LTL";
    {
        "displayName";
        "Litauischer Litas",
            "displayName-count-one";
        "Litauischer Litas",
            "displayName-count-other";
        "Litauische Litas",
            "symbol";
        "LTL",
            "symbol-alt-narrow";
        "Lt";
    }
    "LTT";
    {
        "displayName";
        "Litauischer Talonas",
            "displayName-count-one";
        "Litauische Talonas",

```

```

        "displayName-count-other";
        "Litauische Talonas",
        "symbol";
        "LTT";
    }
    "LUC";
    {
        "displayName";
        "Luxemburgischer Franc (konvertibel)",
        "displayName-count-one";
        "Luxemburgische Franc (konvertibel)",
        "displayName-count-other";
        "Luxemburgische Franc (konvertibel)",
        "symbol";
        "LUC";
    }
    "LUF";
    {
        "displayName";
        "Luxemburgischer Franc",
        "displayName-count-one";
        "Luxemburgische Franc",
        "displayName-count-other";
        "Luxemburgische Franc",
        "symbol";
        "LUF";
    }
    "LUL";
    {
        "displayName";
        "Luxemburgischer Finanz-Franc",
        "displayName-count-one";
        "Luxemburgische Finanz-Franc",
        "displayName-count-other";
        "Luxemburgische Finanz-Franc",
        "symbol";
        "LUL";
    }
    "LVL";
    {
        "displayName";
        "Lettischer Lats",
        "displayName-count-one";
        "Lettischer Lats",
        "displayName-count-other";
        "Lettische Lats",
        "symbol";
        "LVL",
        "symbol-alt-narrow";
        "Ls";
    }
    "LVR";
    {
        "displayName";
        "Lettischer Rubel",
        "displayName-count-one";
        "Lettische Rubel",

```

```

        "displayName-count-other";
        "Lettische Rubel",
        "symbol";
        "LVR";
    }
    "LYD";
    {
        "displayName";
        "Libyscher Dinar",
        "displayName-count-one";
        "Libyscher Dinar",
        "displayName-count-other";
        "Libysche Dinar",
        "symbol";
        "LYD";
    }
    "MAD";
    {
        "displayName";
        "Marokkanischer Dirham",
        "displayName-count-one";
        "Marokkanischer Dirham",
        "displayName-count-other";
        "Marokkanische Dirham",
        "symbol";
        "MAD";
    }
    "MAF";
    {
        "displayName";
        "Marokkanischer Franc",
        "displayName-count-one";
        "Marokkanische Franc",
        "displayName-count-other";
        "Marokkanische Franc",
        "symbol";
        "MAF";
    }
    "MCF";
    {
        "displayName";
        "Monegassischer Franc",
        "displayName-count-one";
        "Monegassischer Franc",
        "displayName-count-other";
        "Monegassische Franc",
        "symbol";
        "MCF";
    }
    "MDC";
    {
        "displayName";
        "Moldau-Cupon",
        "displayName-count-one";
        "Moldau-Cupon",
        "displayName-count-other";
        "Moldau-Cupon",

```

```

        "symbol";
        "MDC";
    }
    "MDL";
    {
        "displayName";
        "Moldau-Leu",
        "displayName-count-one";
        "Moldau-Leu",
        "displayName-count-other";
        "Moldau-Leu",
        "symbol";
        "MDL";
    }
    "MGA";
    {
        "displayName";
        "Madagaskar-Ariary",
        "displayName-count-one";
        "Madagaskar-Ariary",
        "displayName-count-other";
        "Madagaskar-Ariary",
        "symbol";
        "MGA",
        "symbol-alt-narrow";
        "Ar";
    }
    "MGF";
    {
        "displayName";
        "Madagaskar-Franc",
        "displayName-count-one";
        "Madagaskar-Franc",
        "displayName-count-other";
        "Madagaskar-Franc",
        "symbol";
        "MGF";
    }
    "MKD";
    {
        "displayName";
        "Mazedonischer Denar",
        "displayName-count-one";
        "Mazedonischer Denar",
        "displayName-count-other";
        "Mazedonische Denari",
        "symbol";
        "MKD";
    }
    "MKN";
    {
        "displayName";
        "Mazedonischer Denar (1992-1993)",
        "displayName-count-one";
        "Mazedonischer Denar (1992-1993)",
        "displayName-count-other";
        "Mazedonische Denar (1992-1993)",

```

```

        "symbol";
        "MKN";
    }
    "MLF";
    {
        "displayName";
        "Malischer Franc",
        "displayName-count-one";
        "Malische Franc",
        "displayName-count-other";
        "Malische Franc",
        "symbol";
        "MLF";
    }
    "MMK";
    {
        "displayName";
        "Myanmarischer Kyat",
        "displayName-count-one";
        "Myanmarischer Kyat",
        "displayName-count-other";
        "Myanmarische Kyat",
        "symbol";
        "MMK",
        "symbol-alt-narrow";
        "K";
    }
    "MNT";
    {
        "displayName";
        "Mongolischer Tögrög",
        "displayName-count-one";
        "Mongolischer Tögrög",
        "displayName-count-other";
        "Mongolische Tögrög",
        "symbol";
        "MNT",
        "symbol-alt-narrow";
        "₮";
    }
    "MOP";
    {
        "displayName";
        "Macao-Pataca",
        "displayName-count-one";
        "Macao-Pataca",
        "displayName-count-other";
        "Macao-Pataca",
        "symbol";
        "MOP";
    }
    "MRO";
    {
        "displayName";
        "Mauretanischer Ouguiya",
        "displayName-count-one";
        "Mauretanischer Ouguiya",

```

```

        "displayName-count-other";
        "Mauretanische Ouguiya",
        "symbol";
        "MRO";
    }
    "MTL";
    {
        "displayName";
        "Maltesische Lira",
        "displayName-count-one";
        "Maltesische Lira",
        "displayName-count-other";
        "Maltesische Lira",
        "symbol";
        "MTL";
    }
    "MTP";
    {
        "displayName";
        "Maltesisches Pfund",
        "displayName-count-one";
        "Maltesische Pfund",
        "displayName-count-other";
        "Maltesische Pfund",
        "symbol";
        "MTP";
    }
    "MUR";
    {
        "displayName";
        "Mauritius-Rupie",
        "displayName-count-one";
        "Mauritius-Rupie",
        "displayName-count-other";
        "Mauritius-Rupien",
        "symbol";
        "MUR",
        "symbol-alt-narrow";
        "Rs";
    }
    "MVP";
    {
        "displayName";
        "Malediven-Rupie (alt)",
        "displayName-count-one";
        "Malediven-Rupie (alt)",
        "displayName-count-other";
        "Malediven-Rupien (alt)";
    }
    "MVR";
    {
        "displayName";
        "Malediven-Rufiyaa",
        "displayName-count-one";
        "Malediven-Rufiyaa",
        "displayName-count-other";
        "Malediven-Rupien",

```

```

        "symbol";
        "MVR";
    }
    "MWK";
    {
        "displayName";
        "Malawi-Kwacha",
        "displayName-count-one";
        "Malawi-Kwacha",
        "displayName-count-other";
        "Malawi-Kwacha",
        "symbol";
        "MWK";
    }
    "MXN";
    {
        "displayName";
        "Mexikanischer Peso",
        "displayName-count-one";
        "Mexikanischer Peso",
        "displayName-count-other";
        "Mexikanische Pesos",
        "symbol";
        "MX$";
        "symbol-alt-narrow";
        "$";
    }
    "MXP";
    {
        "displayName";
        "Mexikanischer Silber-Peso (1861-1992)",
        "displayName-count-one";
        "Mexikanische Silber-Peso (1861-1992)",
        "displayName-count-other";
        "Mexikanische Silber-Pesos (1861-1992)",
        "symbol";
        "MXP";
    }
    "MXV";
    {
        "displayName";
        "Mexicanischer Unidad de Inversion (UDI)",
        "displayName-count-one";
        "Mexicanischer Unidad de Inversion (UDI)",
        "displayName-count-other";
        "Mexikanische Unidad de Inversion (UDI)",
        "symbol";
        "MXV";
    }
    "MYR";
    {
        "displayName";
        "Malaysischer Ringgit",
        "displayName-count-one";
        "Malaysischer Ringgit",
        "displayName-count-other";
        "Malaysische Ringgit",

```

```

        "symbol";
        "MYR",
        "symbol-alt-narrow";
        "RM";
    }
    "MZE";
    {
        "displayName";
        "Mosambikanischer Escudo",
        "displayName-count-one";
        "Mozambikanische Escudo",
        "displayName-count-other";
        "Mozambikanische Escudo",
        "symbol";
        "MZE";
    }
    "MZM";
    {
        "displayName";
        "Mosambikanischer Metical (1980-2006)",
        "displayName-count-one";
        "Mosambikanischer Metical (1980-2006)",
        "displayName-count-other";
        "Mosambikanische Meticaïs (1980-2006)",
        "symbol";
        "MZM";
    }
    "MZN";
    {
        "displayName";
        "Mosambikanischer Metical",
        "displayName-count-one";
        "Mosambikanischer Metical",
        "displayName-count-other";
        "Mosambikanische Meticaïs",
        "symbol";
        "MZN";
    }
    "NAD";
    {
        "displayName";
        "Namibia-Dollar",
        "displayName-count-one";
        "Namibia-Dollar",
        "displayName-count-other";
        "Namibia-Dollar",
        "symbol";
        "NAD",
        "symbol-alt-narrow";
        "$";
    }
    "NGN";
    {
        "displayName";
        "Nigerianischer Naira",
        "displayName-count-one";
        "Nigerianischer Naira",

```



```

        "displayName-count-other";
        "Nigerianische Naira",
        "symbol";
        "NGN",
        "symbol-alt-narrow";
        "₦";
    }
    "NIC";
    {
        "displayName";
        "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-one";
        "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-other";
        "Nicaraguanische Córdoba (1988-1991)",
        "symbol";
        "NIC";
    }
    "NIO";
    {
        "displayName";
        "Nicaragua-Córdoba",
        "displayName-count-one";
        "Nicaragua-Córdoba",
        "displayName-count-other";
        "Nicaragua-Córdobas",
        "symbol";
        "NIO",
        "symbol-alt-narrow";
        "C$";
    }
    "NLG";
    {
        "displayName";
        "Niederländischer Gulden",
        "displayName-count-one";
        "Niederländischer Gulden",
        "displayName-count-other";
        "Niederländische Gulden",
        "symbol";
        "NLG";
    }
    "NOK";
    {
        "displayName";
        "Norwegische Krone",
        "displayName-count-one";
        "Norwegische Krone",
        "displayName-count-other";
        "Norwegische Kronen",
        "symbol";
        "NOK",
        "symbol-alt-narrow";
        "kr";
    }
    "NPR";
    {

```

```

        "displayName";
        "Nepalesische Rupie",
        "displayName-count-one";
        "Nepalesische Rupie",
        "displayName-count-other";
        "Nepalesische Rupien",
        "symbol";
        "NPR",
        "symbol-alt-narrow";
        "Rs";
    }
    "NZD";
    {
        "displayName";
        "Neuseeland-Dollar",
        "displayName-count-one";
        "Neuseeland-Dollar",
        "displayName-count-other";
        "Neuseeland-Dollar",
        "symbol";
        "NZ$",
        "symbol-alt-narrow";
        "$";
    }
    "OMR";
    {
        "displayName";
        "Omanischer Rial",
        "displayName-count-one";
        "Omanischer Rial",
        "displayName-count-other";
        "Omanische Rials",
        "symbol";
        "OMR";
    }
    "PAB";
    {
        "displayName";
        "Panamaischer Balboa",
        "displayName-count-one";
        "Panamaischer Balboa",
        "displayName-count-other";
        "Panamaische Balboas",
        "symbol";
        "PAB";
    }
    "PEI";
    {
        "displayName";
        "Peruanischer Inti",
        "displayName-count-one";
        "Peruanische Inti",
        "displayName-count-other";
        "Peruanische Inti",
        "symbol";
        "PEI";
    }
}

```

```

    "PEN";
    {
      "displayName";
      "Peruanischer Sol",
      "displayName-count-one";
      "Peruanischer Sol",
      "displayName-count-other";
      "Peruanische Sol",
      "symbol";
      "PEN";
    }
    "PES";
    {
      "displayName";
      "Peruanischer Sol (1863-1965)",
      "displayName-count-one";
      "Peruanischer Sol (1863-1965)",
      "displayName-count-other";
      "Peruanische Sol (1863-1965)",
      "symbol";
      "PES";
    }
    "PGK";
    {
      "displayName";
      "Papua-Neuguineischer Kina",
      "displayName-count-one";
      "Papua-Neuguineischer Kina",
      "displayName-count-other";
      "Papua-Neuguineische Kina",
      "symbol";
      "PGK";
    }
    "PHP";
    {
      "displayName";
      "Philippinischer Peso",
      "displayName-count-one";
      "Philippinischer Peso",
      "displayName-count-other";
      "Philippinische Pesos",
      "symbol";
      "PHP",
      "symbol-alt-narrow";
      "₱";
    }
    "PKR";
    {
      "displayName";
      "Pakistanische Rupie",
      "displayName-count-one";
      "Pakistanische Rupie",
      "displayName-count-other";
      "Pakistanische Rupien",
      "symbol";
      "PKR",
      "symbol-alt-narrow";
    }

```

```

        "Rs";
    }
    "PLN";
    {
        "displayName";
        "Polnischer Złoty",
        "displayName-count-one";
        "Polnischer Złoty",
        "displayName-count-other";
        "Polnische Złoty",
        "symbol";
        "PLN",
        "symbol-alt-narrow";
        "zł";
    }
    "PLZ";
    {
        "displayName";
        "Polnischer Zloty (1950-1995)",
        "displayName-count-one";
        "Polnischer Zloty (1950-1995)",
        "displayName-count-other";
        "Polnische Zloty (1950-1995)",
        "symbol";
        "PLZ";
    }
    "PTE";
    {
        "displayName";
        "Portugiesischer Escudo",
        "displayName-count-one";
        "Portugiesische Escudo",
        "displayName-count-other";
        "Portugiesische Escudo",
        "symbol";
        "PTE";
    }
    "PYG";
    {
        "displayName";
        "Paraguayischer Guaraní",
        "displayName-count-one";
        "Paraguayischer Guaraní",
        "displayName-count-other";
        "Paraguayische Guaraníes",
        "symbol";
        "PYG",
        "symbol-alt-narrow";
        "₲";
    }
    "QAR";
    {
        "displayName";
        "Katar-Riyal",
        "displayName-count-one";
        "Katar-Riyal",
        "displayName-count-other";
    }

```

```

        "Katar-Riyal",
        "symbol";
        "QAR";
    }
    "RHD";
    {
        "displayName";
        "Rhodesischer Dollar",
        "displayName-count-one";
        "Rhodesische Dollar",
        "displayName-count-other";
        "Rhodesische Dollar",
        "symbol";
        "RHD";
    }
    "ROL";
    {
        "displayName";
        "Rumänischer Leu (1952-2006)",
        "displayName-count-one";
        "Rumänischer Leu (1952-2006)",
        "displayName-count-other";
        "Rumänische Leu (1952-2006)",
        "symbol";
        "ROL";
    }
    "RON";
    {
        "displayName";
        "Rumänischer Leu",
        "displayName-count-one";
        "Rumänischer Leu",
        "displayName-count-other";
        "Rumänische Leu",
        "symbol";
        "RON",
        "symbol-alt-narrow";
        "L";
    }
    "RSD";
    {
        "displayName";
        "Serbischer Dinar",
        "displayName-count-one";
        "Serbischer Dinar",
        "displayName-count-other";
        "Serbische Dinaren",
        "symbol";
        "RSD";
    }
    "RUB";
    {
        "displayName";
        "Russischer Rubel",
        "displayName-count-one";
        "Russischer Rubel",
        "displayName-count-other";
    }

```

```

        "Russische Rubel",
        "symbol";
        "RUB",
        "symbol-alt-narrow";
        "₽";
    }
    "RUR";
    {
        "displayName";
        "Russischer Rubel (1991-1998)",
        "displayName-count-one";
        "Russischer Rubel (1991-1998)",
        "displayName-count-other";
        "Russische Rubel (1991-1998)",
        "symbol";
        "RUR",
        "symbol-alt-narrow";
        "p.";
    }
    "RWF";
    {
        "displayName";
        "Ruanda-Franc",
        "displayName-count-one";
        "Ruanda-Franc",
        "displayName-count-other";
        "Ruanda-Francis",
        "symbol";
        "RWF",
        "symbol-alt-narrow";
        "F.Rw";
    }
    "SAR";
    {
        "displayName";
        "Saudi-Rial",
        "displayName-count-one";
        "Saudi-Rial",
        "displayName-count-other";
        "Saudi-Rial",
        "symbol";
        "SAR";
    }
    "SBD";
    {
        "displayName";
        "Salomonen-Dollar",
        "displayName-count-one";
        "Salomonen-Dollar",
        "displayName-count-other";
        "Salomonen-Dollar",
        "symbol";
        "SBD",
        "symbol-alt-narrow";
        "$";
    }
    "SCR";

```

```

    {
      "displayName";
      "Seychellen-Rupie",
      "displayName-count-one";
      "Seychellen-Rupie",
      "displayName-count-other";
      "Seychellen-Rupien",
      "symbol";
      "SCR";
    }
    "SDD";
    {
      "displayName";
      "Sudanesischer Dinar (1992-2007)",
      "displayName-count-one";
      "Sudanesischer Dinar (1992-2007)",
      "displayName-count-other";
      "Sudanesische Dinar (1992-2007)",
      "symbol";
      "SDD";
    }
    "SDG";
    {
      "displayName";
      "Sudanesisches Pfund",
      "displayName-count-one";
      "Sudanesisches Pfund",
      "displayName-count-other";
      "Sudanesische Pfund",
      "symbol";
      "SDG";
    }
    "SDP";
    {
      "displayName";
      "Sudanesisches Pfund (1957-1998)",
      "displayName-count-one";
      "Sudanesisches Pfund (1957-1998)",
      "displayName-count-other";
      "Sudanesische Pfund (1957-1998)",
      "symbol";
      "SDP";
    }
    "SEK";
    {
      "displayName";
      "Schwedische Krone",
      "displayName-count-one";
      "Schwedische Krone",
      "displayName-count-other";
      "Schwedische Kronen",
      "symbol";
      "SEK",
      "symbol-alt-narrow";
      "kr";
    }
    "SGD";

```

```

    {
      "displayName";
      "Singapur-Dollar",
        "displayName-count-one";
      "Singapur-Dollar",
        "displayName-count-other";
      "Singapur-Dollar",
        "symbol";
      "SGD",
        "symbol-alt-narrow";
      "$";
    }
    "SHP";
    {
      "displayName";
      "St. Helena-Pfund",
        "displayName-count-one";
      "St. Helena-Pfund",
        "displayName-count-other";
      "St. Helena-Pfund",
        "symbol";
      "SHP",
        "symbol-alt-narrow";
      "£";
    }
    "SIT";
    {
      "displayName";
      "Slowenischer Tolar",
        "displayName-count-one";
      "Slowenischer Tolar",
        "displayName-count-other";
      "Slowenische Tolar",
        "symbol";
      "SIT";
    }
    "SKK";
    {
      "displayName";
      "Slowakische Krone",
        "displayName-count-one";
      "Slowakische Kronen",
        "displayName-count-other";
      "Slowakische Kronen",
        "symbol";
      "SKK";
    }
    "SLL";
    {
      "displayName";
      "Sierra-leonischer Leone",
        "displayName-count-one";
      "Sierra-leonischer Leone",
        "displayName-count-other";
      "Sierra-leonische Leones",
        "symbol";
      "SLL";
    }
  }

```



```

    }
    "SOS";
    {
        "displayName";
        "Somalia-Schilling",
        "displayName-count-one";
        "Somalia-Schilling",
        "displayName-count-other";
        "Somalia-Schilling",
        "symbol";
        "SOS";
    }
    "SRD";
    {
        "displayName";
        "Suriname-Dollar",
        "displayName-count-one";
        "Suriname-Dollar",
        "displayName-count-other";
        "Suriname-Dollar",
        "symbol";
        "SRD",
        "symbol-alt-narrow";
        "$";
    }
    "SRG";
    {
        "displayName";
        "Suriname Gulden",
        "displayName-count-one";
        "Suriname-Gulden",
        "displayName-count-other";
        "Suriname-Gulden",
        "symbol";
        "SRG";
    }
    "SSP";
    {
        "displayName";
        "Südsudanesisches Pfund",
        "displayName-count-one";
        "Südsudanesisches Pfund",
        "displayName-count-other";
        "Südsudanesische Pfund",
        "symbol";
        "SSP",
        "symbol-alt-narrow";
        "£";
    }
    "STD";
    {
        "displayName";
        "São-toméischer Dobra",
        "displayName-count-one";
        "São-toméischer Dobra",
        "displayName-count-other";
        "São-toméische Dobra",

```

```

        "symbol";
        "STD",
        "symbol-alt-narrow";
        "Db";
    }
    "SUR";
    {
        "displayName";
        "Sowjetischer Rubel",
        "displayName-count-one";
        "Sowjetische Rubel",
        "displayName-count-other";
        "Sowjetische Rubel",
        "symbol";
        "SUR";
    }
    "SVC";
    {
        "displayName";
        "El Salvador Colon",
        "displayName-count-one";
        "El Salvador-Colon",
        "displayName-count-other";
        "El Salvador-Colon",
        "symbol";
        "SVC";
    }
    "SYP";
    {
        "displayName";
        "Syrisches Pfund",
        "displayName-count-one";
        "Syrisches Pfund",
        "displayName-count-other";
        "Syrische Pfund",
        "symbol";
        "SYP",
        "symbol-alt-narrow";
        "SYP";
    }
    "SZL";
    {
        "displayName";
        "Swasiländischer Lilangeni",
        "displayName-count-one";
        "Swasiländischer Lilangeni",
        "displayName-count-other";
        "Swasiländische Emalangeni",
        "symbol";
        "SZL";
    }
    "THB";
    {
        "displayName";
        "Thailändischer Baht",
        "displayName-count-one";
        "Thailändischer Baht",

```

```

        "displayName-count-other";
        "Thailändische Baht",
        "symbol";
        "฿",
        "symbol-alt-narrow";
        "฿";
    }
    "TJR";
    {
        "displayName";
        "Tadschikistan Rubel",
        "displayName-count-one";
        "Tadschikistan-Rubel",
        "displayName-count-other";
        "Tadschikistan-Rubel",
        "symbol";
        "TJR";
    }
    "TJS";
    {
        "displayName";
        "Tadschikistan-Somoni",
        "displayName-count-one";
        "Tadschikistan-Somoni",
        "displayName-count-other";
        "Tadschikistan-Somoni",
        "symbol";
        "TJS";
    }
    "TMM";
    {
        "displayName";
        "Turkmenistan-Manat (1993-2009)",
        "displayName-count-one";
        "Turkmenistan-Manat (1993-2009)",
        "displayName-count-other";
        "Turkmenistan-Manat (1993-2009)",
        "symbol";
        "TMM";
    }
    "TMT";
    {
        "displayName";
        "Turkmenistan-Manat",
        "displayName-count-one";
        "Turkmenistan-Manat",
        "displayName-count-other";
        "Turkmenistan-Manat",
        "symbol";
        "TMT";
    }
    "TND";
    {
        "displayName";
        "Tunesischer Dinar",
        "displayName-count-one";
        "Tunesischer Dinar",

```

```

        "displayName-count-other";
        "Tunesische Dinar",
        "symbol";
        "TND";
    }
    "TOP";
    {
        "displayName";
        "Tongaischer Pa'anga",
        "displayName-count-one";
        "Tongaischer Pa'anga",
        "displayName-count-other";
        "Tongaische Pa'anga",
        "symbol";
        "TOP",
        "symbol-alt-narrow";
        "T$";
    }
    "TPE";
    {
        "displayName";
        "Timor-Escudo",
        "displayName-count-one";
        "Timor-Escudo",
        "displayName-count-other";
        "Timor-Escudo",
        "symbol";
        "TPE";
    }
    "TRL";
    {
        "displayName";
        "Türkische Lira (1922-2005)",
        "displayName-count-one";
        "Türkische Lira (1922-2005)",
        "displayName-count-other";
        "Türkische Lira (1922-2005)",
        "symbol";
        "TRL";
    }
    "TRY";
    {
        "displayName";
        "Türkische Lira",
        "displayName-count-one";
        "Türkische Lira",
        "displayName-count-other";
        "Türkische Lira",
        "symbol";
        "TRY",
        "symbol-alt-narrow";
        "₺",
        "symbol-alt-variant";
        "TL";
    }
    "TTD";
    {

```

```

        "displayName";
        "Trinidad und Tobago-Dollar",
            "displayName-count-one";
        "Trinidad und Tobago-Dollar",
            "displayName-count-other";
        "Trinidad und Tobago-Dollar",
            "symbol";
        "TTD",
            "symbol-alt-narrow";
        "$";
    }
    "TWD";
    {
        "displayName";
        "Neuer Taiwan-Dollar",
            "displayName-count-one";
        "Neuer Taiwan-Dollar",
            "displayName-count-other";
        "Neue Taiwan-Dollar",
            "symbol";
        "NT$";
            "symbol-alt-narrow";
        "NT$";
    }
    "TZS";
    {
        "displayName";
        "Tansania-Schilling",
            "displayName-count-one";
        "Tansania-Schilling",
            "displayName-count-other";
        "Tansania-Schilling",
            "symbol";
        "TZS";
    }
    "UAH";
    {
        "displayName";
        "Ukrainische Hrywnja",
            "displayName-count-one";
        "Ukrainische Hrywnja",
            "displayName-count-other";
        "Ukrainische Hrywen",
            "symbol";
        "UAH",
            "symbol-alt-narrow";
        "₴";
    }
    "UAK";
    {
        "displayName";
        "Ukrainischer Karbovanetz",
            "displayName-count-one";
        "Ukrainische Karbovanetz",
            "displayName-count-other";
        "Ukrainische Karbovanetz",
            "symbol";
    }

```

```

        "UAK";
    }
    "UGS";
    {
        "displayName";
        "Uganda-Schilling (1966-1987)",
        "displayName-count-one";
        "Uganda-Schilling (1966-1987)",
        "displayName-count-other";
        "Uganda-Schilling (1966-1987)",
        "symbol";
        "UGS";
    }
    "UGX";
    {
        "displayName";
        "Uganda-Schilling",
        "displayName-count-one";
        "Uganda-Schilling",
        "displayName-count-other";
        "Uganda-Schilling",
        "symbol";
        "UGX";
    }
    "USD";
    {
        "displayName";
        "US-Dollar",
        "displayName-count-one";
        "US-Dollar",
        "displayName-count-other";
        "US-Dollar",
        "symbol";
        "$",
        "symbol-alt-narrow";
        "$";
    }
    "USN";
    {
        "displayName";
        "US Dollar (Nächster Tag)",
        "displayName-count-one";
        "US-Dollar (Nächster Tag)",
        "displayName-count-other";
        "US-Dollar (Nächster Tag)",
        "symbol";
        "USN";
    }
    "USS";
    {
        "displayName";
        "US Dollar (Gleicher Tag)",
        "displayName-count-one";
        "US-Dollar (Gleicher Tag)",
        "displayName-count-other";
        "US-Dollar (Gleicher Tag)",
        "symbol";
    }

```

```

        "USS";
    }
    "UYI";
    {
        "displayName";
        "Uruguayischer Peso (Indexierte Rechnungseinheiten)",
        "displayName-count-one";
        "Uruguayischer Peso (Indexierte Rechnungseinheiten)",
        "displayName-count-other";
        "Uruguayische Pesos (Indexierte Rechnungseinheiten)",
        "symbol";
        "UYI";
    }
    "UYP";
    {
        "displayName";
        "Uruguayischer Peso (1975-1993)",
        "displayName-count-one";
        "Uruguayischer Peso (1975-1993)",
        "displayName-count-other";
        "Uruguayische Pesos (1975-1993)",
        "symbol";
        "UYP";
    }
    "UYU";
    {
        "displayName";
        "Uruguayischer Peso",
        "displayName-count-one";
        "Uruguayischer Peso",
        "displayName-count-other";
        "Uruguayische Pesos",
        "symbol";
        "UYU",
        "symbol-alt-narrow";
        "$";
    }
    "UZS";
    {
        "displayName";
        "Usbekistan-Sum",
        "displayName-count-one";
        "Usbekistan-Sum",
        "displayName-count-other";
        "Usbekistan-Sum",
        "symbol";
        "UZS";
    }
    "VEB";
    {
        "displayName";
        "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-one";
        "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-other";
        "Venezolanische Bolívares (1871-2008)",
        "symbol";
    }

```

```

        "VEB";
    }
    "VEF";
    {
        "displayName";
        "Venezolanischer Bolívar",
        "displayName-count-one";
        "Venezolanischer Bolívar",
        "displayName-count-other";
        "Venezolanische Bolívares",
        "symbol";
        "VEF",
        "symbol-alt-narrow";
        "Bs";
    }
    "VND";
    {
        "displayName";
        "Vietnamesischer Dong",
        "displayName-count-one";
        "Vietnamesischer Dong",
        "displayName-count-other";
        "Vietnamesische Dong",
        "symbol";
        "₫",
        "symbol-alt-narrow";
        "₫";
    }
    "VNN";
    {
        "displayName";
        "Vietnamesischer Dong (1978-1985)",
        "displayName-count-one";
        "Vietnamesischer Dong (1978-1985)",
        "displayName-count-other";
        "Vietnamesische Dong (1978-1985)",
        "symbol";
        "VNN";
    }
    "VUV";
    {
        "displayName";
        "Vanuatu-Vatu",
        "displayName-count-one";
        "Vanuatu-Vatu",
        "displayName-count-other";
        "Vanuatu-Vatu",
        "symbol";
        "VUV";
    }
    "WST";
    {
        "displayName";
        "Samoanischer Tala",
        "displayName-count-one";
        "Samoanischer Tala",
        "displayName-count-other";
    }

```



```

        "Samoanische Tala",
        "symbol";
        "WST";
    }
    "XAF";
    {
        "displayName";
        "CFA-Franc (BEAC) ",
        "displayName-count-one";
        "CFA-Franc (BEAC) ",
        "displayName-count-other";
        "CFA-Franc (BEAC) ",
        "symbol";
        "FCFA";
    }
    "XAG";
    {
        "displayName";
        "Unze Silber",
        "displayName-count-one";
        "Unze Silber",
        "displayName-count-other";
        "Unzen Silber",
        "symbol";
        "XAG";
    }
    "XAU";
    {
        "displayName";
        "Unze Gold",
        "displayName-count-one";
        "Unze Gold",
        "displayName-count-other";
        "Unzen Gold",
        "symbol";
        "XAU";
    }
    "XBA";
    {
        "displayName";
        "Europäische Rechnungseinheit",
        "displayName-count-one";
        "Europäische Rechnungseinheiten",
        "displayName-count-other";
        "Europäische Rechnungseinheiten",
        "symbol";
        "XBA";
    }
    "XBB";
    {
        "displayName";
        "Europäische Währungseinheit (XBB) ",
        "displayName-count-one";
        "Europäische Währungseinheiten (XBB) ",
        "displayName-count-other";
        "Europäische Währungseinheiten (XBB) ",
        "symbol";
    }

```

```

        "XBB";
    }
    "XBC";
    {
        "displayName";
        "Europäische Rechnungseinheit (XBC)",
        "displayName-count-one";
        "Europäische Rechnungseinheiten (XBC)",
        "displayName-count-other";
        "Europäische Rechnungseinheiten (XBC)",
        "symbol";
        "XBC";
    }
    "XBD";
    {
        "displayName";
        "Europäische Rechnungseinheit (XBD)",
        "displayName-count-one";
        "Europäische Rechnungseinheiten (XBD)",
        "displayName-count-other";
        "Europäische Rechnungseinheiten (XBD)",
        "symbol";
        "XBD";
    }
    "XCD";
    {
        "displayName";
        "Ostkaribischer Dollar",
        "displayName-count-one";
        "Ostkaribischer Dollar",
        "displayName-count-other";
        "Ostkaribische Dollar",
        "symbol";
        "EC$",
        "symbol-alt-narrow";
        "$";
    }
    "XDR";
    {
        "displayName";
        "Sonderziehungsrechte",
        "displayName-count-one";
        "Sonderziehungsrechte",
        "displayName-count-other";
        "Sonderziehungsrechte",
        "symbol";
        "XDR";
    }
    "XEU";
    {
        "displayName";
        "Europäische Währungseinheit (XEU)",
        "displayName-count-one";
        "Europäische Währungseinheiten (XEU)",
        "displayName-count-other";
        "Europäische Währungseinheiten (XEU)",
        "symbol";
    }

```

```

        "XEU";
    }
    "XFO";
    {
        "displayName";
        "Französischer Gold-Franc",
        "displayName-count-one";
        "Französische Gold-Franc",
        "displayName-count-other";
        "Französische Gold-Franc",
        "symbol";
        "XFO";
    }
    "XFU";
    {
        "displayName";
        "Französischer UIC-Franc",
        "displayName-count-one";
        "Französische UIC-Franc",
        "displayName-count-other";
        "Französische UIC-Franc",
        "symbol";
        "XFU";
    }
    "XOF";
    {
        "displayName";
        "CFA-Franc (BCEAO)",
        "displayName-count-one";
        "CFA-Franc (BCEAO)",
        "displayName-count-other";
        "CFA-Francs (BCEAO)",
        "symbol";
        "CFA";
    }
    "XPD";
    {
        "displayName";
        "Unze Palladium",
        "displayName-count-one";
        "Unze Palladium",
        "displayName-count-other";
        "Unzen Palladium",
        "symbol";
        "XPD";
    }
    "XPF";
    {
        "displayName";
        "CFP-Franc",
        "displayName-count-one";
        "CFP-Franc",
        "displayName-count-other";
        "CFP-Franc",
        "symbol";
        "CFPF";
    }
}

```

```

    "XPT";
    {
        "displayName";
        "Unze Platin",
            "displayName-count-one";
        "Unze Platin",
            "displayName-count-other";
        "Unzen Platin",
            "symbol";
        "XPT";
    }
    "XRE";
    {
        "displayName";
        "RINET Funds",
            "displayName-count-one";
        "RINET Funds",
            "displayName-count-other";
        "RINET Funds",
            "symbol";
        "XRE";
    }
    "XSU";
    {
        "displayName";
        "SUCRE",
            "displayName-count-one";
        "SUCRE",
            "displayName-count-other";
        "SUCRE",
            "symbol";
        "XSU";
    }
    "XTS";
    {
        "displayName";
        "Testwährung",
            "displayName-count-one";
        "Testwährung",
            "displayName-count-other";
        "Testwährung",
            "symbol";
        "XTS";
    }
    "XUA";
    {
        "displayName";
        "Rechnungseinheit der AfEB",
            "displayName-count-one";
        "Rechnungseinheit der AfEB",
            "displayName-count-other";
        "Rechnungseinheiten der AfEB",
            "symbol";
        "XUA";
    }
    "XXX";
    {

```

```

        "displayName";
        "Unbekannte Währung",
        "displayName-count-one";
        "(unbekannte Währung)",
        "displayName-count-other";
        "(unbekannte Währung)",
        "symbol";
        "XXX";
    }
    "YDD";
    {
        "displayName";
        "Jemen-Dinar",
        "displayName-count-one";
        "Jemen-Dinar",
        "displayName-count-other";
        "Jemen-Dinar",
        "symbol";
        "YDD";
    }
    "YER";
    {
        "displayName";
        "Jemen-Rial",
        "displayName-count-one";
        "Jemen-Rial",
        "displayName-count-other";
        "Jemen-Rial",
        "symbol";
        "YER";
    }
    "YUD";
    {
        "displayName";
        "Jugoslawischer Dinar (1966-1990)",
        "displayName-count-one";
        "Jugoslawischer Dinar (1966-1990)",
        "displayName-count-other";
        "Jugoslawische Dinar (1966-1990)",
        "symbol";
        "YUD";
    }
    "YUM";
    {
        "displayName";
        "Jugoslawischer Neuer Dinar (1994-2002)",
        "displayName-count-one";
        "Jugoslawischer Neuer Dinar (1994-2002)",
        "displayName-count-other";
        "Jugoslawische Neue Dinar (1994-2002)",
        "symbol";
        "YUM";
    }
    "YUN";
    {
        "displayName";
        "Jugoslawischer Dinar (konvertibel)",

```

```

        "displayName-count-one";
        "Jugoslawische Dinar (konvertibel)",
        "displayName-count-other";
        "Jugoslawische Dinar (konvertibel)",
        "symbol";
        "YUN";
    }
    "YUR";
    {
        "displayName";
        "Jugoslawischer reformierter Dinar (1992-1993)",
        "displayName-count-one";
        "Jugoslawischer reformierter Dinar (1992-1993)",
        "displayName-count-other";
        "Jugoslawische reformierte Dinar (1992-1993)",
        "symbol";
        "YUR";
    }
    "ZAL";
    {
        "displayName";
        "Südafrikanischer Rand (Finanz)",
        "displayName-count-one";
        "Südafrikanischer Rand (Finanz)",
        "displayName-count-other";
        "Südafrikanischer Rand (Finanz)",
        "symbol";
        "ZAL";
    }
    "ZAR";
    {
        "displayName";
        "Südafrikanischer Rand",
        "displayName-count-one";
        "Südafrikanischer Rand",
        "displayName-count-other";
        "Südafrikanische Rand",
        "symbol";
        "ZAR",
        "symbol-alt-narrow";
        "R";
    }
    "ZMK";
    {
        "displayName";
        "Kwacha (1968-2012)",
        "displayName-count-one";
        "Kwacha (1968-2012)",
        "displayName-count-other";
        "Kwacha (1968-2012)",
        "symbol";
        "ZMK";
    }
    "ZMW";
    {
        "displayName";
        "Kwacha",

```

```

        "displayName-count-one";
        "Kwacha",
        "displayName-count-other";
        "Kwacha",
        "symbol";
        "ZMW",
        "symbol-alt-narrow";
        "K";
    }
    "ZRN";
    {
        "displayName";
        "Zaire-Neuer Zaïre (1993-1998)",
        "displayName-count-one";
        "Zaire-Neuer Zaïre (1993-1998)",
        "displayName-count-other";
        "Zaire-Neue Zaïre (1993-1998)",
        "symbol";
        "ZRN";
    }
    "ZRZ";
    {
        "displayName";
        "Zaire-Zaïre (1971-1993)",
        "displayName-count-one";
        "Zaire-Zaïre (1971-1993)",
        "displayName-count-other";
        "Zaire-Zaïre (1971-1993)",
        "symbol";
        "ZRZ";
    }
    "ZWD";
    {
        "displayName";
        "Simbabwe-Dollar (1980-2008)",
        "displayName-count-one";
        "Simbabwe-Dollar (1980-2008)",
        "displayName-count-other";
        "Simbabwe-Dollar (1980-2008)",
        "symbol";
        "ZWD";
    }
    "ZWL";
    {
        "displayName";
        "Simbabwe-Dollar (2009)",
        "displayName-count-one";
        "Simbabwe-Dollar (2009)",
        "displayName-count-other";
        "Simbabwe-Dollar (2009)",
        "symbol";
        "ZWL";
    }
    "ZWR";
    {
        "displayName";
        "Simbabwe-Dollar (2008)",

```

```
        "displayName-count-one";  
        "Simbabwe-Dollar (2008)",  
        "displayName-count-other";  
        "Simbabwe-Dollar (2008)",  
        "symbol";  
    "ZWR";  
}  
  
}  
  
}
```

INDEX.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// Load the L10n, loadCldr from ej2-base
import { loadCldr, L10n } from '@syncfusion/ej2-base';
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
// load the CLDR data files.
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as timeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames);
L10n.load({
    'de': {
        'datepicker': { placeholder: 'Wählen Sie ein Datum aus',
            today: 'heute' }
    }
});
// import the datepickercomponent
class App extends React.Component {
    render() {
        return <DatePickerComponent id="datepicker" locale='de'/>;
    }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// Load the L10n, loadCldr from ej2-base
import { loadCldr, L10n } from '@syncfusion/ej2-base';
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
// load the CLDR data files.
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as timeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames);
L10n.load({
```



```

    'de': {
      'datepicker': { placeholder: 'Wählen Sie ein Datum aus',
        today: 'heute' }
    }
  });
  // import the datepickercomponent
  class App extends React.Component<{}, {}> {
    render() {
      return <DatePickerComponent id="datepicker" locale='de'/>;
    }
  }
  ReactDOM.render(<App />, document.getElementById('element'));

```

NUMBERINGSYSTEMS.JSON

```

{
  "supplemental": {
    "version": {
      "_number": "$Revision: 12732 $",
      "_unicodeVersion": "9.0.0",
      "_cldrVersion": "31"
    },
    "numberingSystems": {
      "adlm": {
        "_digits": "ᲐᲑᲒᲓᲔᲕᲖᲗᲘᲙᲚᲛᲜᲝᲞᲟᲠᲡᲢᲣᲤᲥᲦᲧᲨᲩᲪᲫᲬᲭᲮᲯᲰᲱᲲᲳᲴᲵᲶᲷᲸᲹᲺ᲻᲼ᲽᲾᲿ",
        "_type": "numeric"
      },
      "ahom": {
        "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍᱎᱏ᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱹᱺᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱹᱺᱛ",
        "_type": "numeric"
      },
      "arab": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "arabext": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "armn": {
        "_rules": "armenian-upper",
        "_type": "algorithmic"
      },
      "armnlow": {
        "_rules": "armenian-lower",
        "_type": "algorithmic"
      },
      "bali": {
        "_digits": "᭐ᭀᭁᭂᭃ᭄ᭅᭆᭇᭈᭉᭊᭋᭌ᭍᭎᭏᭐᭑᭒᭓᭔᭕᭖᭗᭘᭙᭚᭛᭜᭝᭞᭟᭠᭡᭢᭣᭤᭥᭦᭧᭨᭩᭪᭬᭫᭭᭮᭯᭰᭱᭲᭳᭴᭵᭶᭷᭸᭹᭺᭻᭼᭾᭿",
        "_type": "numeric"
      },
      "beng": {
        "_digits": "০১২৩৪৫৬৭৮৯",
        "_type": "numeric"
      },
      "bhks": {

```

```
{
  "_digits": "□□□□□□□□",
  "_type": "numeric"
},
"brah": {
  "_digits": "·\\۳۴۵۶۷۸",
  "_type": "numeric"
},
"cakm": {
  "_digits": "ᲠᲡᲢᲣᲤᲥᲦᲧ",
  "_type": "numeric"
},
"cham": {
  "_digits": "ꨀꨁꨂꨃꨄꨅꨆꨇ",
  "_type": "numeric"
},
"cyrl": {
  "_rules": "cyrillic-lower",
  "_type": "algorithmic"
},
"deva": {
  "_digits": "ॐ॒॑॓॔ॕॖॗक़ख़",
  "_type": "numeric"
},
"ethi": {
  "_rules": "ethiopic",
  "_type": "algorithmic"
},
"fullwide": {
  "_digits": "0 1 2 3 4 5 6 7 8 9",
  "_type": "numeric"
},
"geor": {
  "_rules": "georgian",
  "_type": "algorithmic"
},
"grek": {
  "_rules": "greek-upper",
  "_type": "algorithmic"
},
"greklow": {
  "_rules": "greek-lower",
  "_type": "algorithmic"
},
"gujr": {
  "_digits": "ૐ૑૒૓૔૕૖૗૘૙",
  "_type": "numeric"
},
"guru": {
  "_digits": "੐ੑ੒੓੔੕੖੗੘ਖ਼",
  "_type": "numeric"
},
"hanidays": {
  "_rules": "zh/SpelloutRules/spellout-numbering-days",
  "_type": "algorithmic"
}
```

```

"hanidec": {
  "_digits": "〇一二三四五六七八九",
  "_type": "numeric"
},
"hans": {
  "_rules": "zh/SpelloutRules/spellout-cardinal",
  "_type": "algorithmic"
},
"hansfin": {
  "_rules": "zh/SpelloutRules/spellout-cardinal-financial",
  "_type": "algorithmic"
},
"hant": {
  "_rules": "zh_Hant/SpelloutRules/spellout-cardinal",
  "_type": "algorithmic"
},
"hantfin": {
  "_rules": "zh_Hant/SpelloutRules/spellout-cardinal-financial",
  "_type": "algorithmic"
},
"hebr": {
  "_rules": "hebrew",
  "_type": "algorithmic"
},
"hmng": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"java": {
  "_digits": "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉᐊᐋᐌᐍᐎᐏᐐᐑᐒᐓᐔᐕᐖᐗᐘᐙᐚᐛᐜᐝᐞᐟᐠᐡᐢᐣᐤᐥᐦᐧᐨᐩᐪᐫᐬᐭᐮᐯᐰᐱᐲᐳᐴᐵᐶᐷᐸᐹᐺᐻᐼᐽᐾᐿ",
  "_type": "numeric"
},
"jpan": {
  "_rules": "ja/SpelloutRules/spellout-cardinal",
  "_type": "algorithmic"
},
"jpanfin": {
  "_rules": "ja/SpelloutRules/spellout-cardinal-financial",
  "_type": "algorithmic"
},
"kali": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"khmr": {
  "_digits": "០១២៣៤៥៦៧៨៩",
  "_type": "numeric"
},
"knda": {
  "_digits": "೦೧೨೩೪೫೬೭೮೯",
  "_type": "numeric"
},
"lana": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
}

```

```
},
"lanatham": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"laoo": {
  "_digits": "໐໑໒໓໔໕໖໗໘໑",
  "_type": "numeric"
},
"latn": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"lepc": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"limb": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"mathbold": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathdbl": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathmono": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathsanb": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathsans": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mlym": {
  "_digits": "൦൧൨൩൪൫൬൭൮൯",
  "_type": "numeric"
},
"modi": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"mong": {
  "_digits": "᠐᠑᠒᠓᠐ᠠᠨᠨᠠᠭ",
  "_type": "numeric"
},
"mroo": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
}
```

```

},
"mtei": {
  "_digits": "ᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐ",
  "_type": "numeric"
},
"mymr": {
  "_digits": "၀၁၂၃၄၅၆၇၈",
  "_type": "numeric"
},
"mymrshan": {
  "_digits": "၀၁၂၃၄၅၆၇၈",
  "_type": "numeric"
},
"mymrtlng": {
  "_digits": "၀၁၂၃၄၅၆၇၈",
  "_type": "numeric"
},
"newa": {
  "_digits": "ᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐ",
  "_type": "numeric"
},
"nkoo": {
  "_digits": "ᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐ",
  "_type": "numeric"
},
"olck": {
  "_digits": "ᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐ",
  "_type": "numeric"
},
"orya": {
  "_digits": "୦୧୨୩୪୫୬୭୮୯",
  "_type": "numeric"
},
"osma": {
  "_digits": "ᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐ",
  "_type": "numeric"
},
"roman": {
  "_rules": "roman-upper",
  "_type": "algorithmic"
},
"romanlow": {
  "_rules": "roman-lower",
  "_type": "algorithmic"
},
"saur": {
  "_digits": "ᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐ",
  "_type": "numeric"
},
"shrd": {
  "_digits": "ᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐᐐ",
  "_type": "numeric"
},
"sind": {

```

```

    "_digits": "□□□□□□□□",
    "_type": "numeric"
  },
  "sinh": {
    "_digits": "෦෧෨෩෪෫෬෭෮෯",
    "_type": "numeric"
  },
  "sora": {
    "_digits": "୦୧୨୩୪୫୬୭୮୯",
    "_type": "numeric"
  },
  "sund": {
    "_digits": "□□□□□□□□",
    "_type": "numeric"
  },
  "takr": {
    "_digits": "□□□□□□□□",
    "_type": "numeric"
  },
  "taluk": {
    "_digits": "೦೧೨೩೪೫೬೭೮೯",
    "_type": "numeric"
  },
  "taml": {
    "_rules": "tamil",
    "_type": "algorithmic"
  },
  "tamldec": {
    "_digits": "௦௧௨௩௪௫௬௭௮௯",
    "_type": "numeric"
  },
  "telu": {
    "_digits": "୦୧୨୩୪୫୬୭୮୯",
    "_type": "numeric"
  },
  "thai": {
    "_digits": "๐๑๒๓๔๕๖๗๘๙",
    "_type": "numeric"
  },
  "tibet": {
    "_digits": "༠༡༢༣༤༥༦༧༨༩",
    "_type": "numeric"
  },
  "tirh": {
    "_digits": "□□□□□□□□",
    "_type": "numeric"
  },
  "vaih": {
    "_digits": "໐໑໒໓໔໕໖໗໘໙",
    "_type": "numeric"
  },
  "wara": {
    "_digits": "□□□□□□□□",
    "_type": "numeric"
  }
}

```

```

    }
  }
}

```

NUMBERINGSYSTEMS.JSX

```

{
  "supplemental";
  {
    "version";
    {
      "_number";
      "$Revision: 12732 $",
      "_unicodeVersion";
      "9.0.0",
      "_cldrVersion";
      "31";
    }
    "numberingSystems";
    {
      "adlm";
      {
        "_digits";
        "ᲐᲑᲔᲕᲗᲘᲙᲚᲛᲞᲟᲠᲡᲢᲣᲤᲥᲦᲧᲨᲩᲪᲫᲬᲭᲮᲯᲰᲱᲲᲳᲴᲵᲶᲷᲸᲹᲺ᲻᲼ᲽᲾᲿ",
        "_type";
        "numeric";
      }
      "ahom";
      {
        "_digits";
        "ᩌᩍᩎᩏᩐᩑᩒᩓᩔᩕᩖᩗᩘᩙᩚᩛᩜᩝᩞ᩟᩠ᩡᩢᩣᩤᩥᩦᩧᩨᩩᩪᩫᩬᩭᩮᩯᩰᩱᩲᩳᩴ᩵᩶᩷᩸᩹᩺᩻᩼᩽᩾᩿",
        "_type";
        "numeric";
      }
      "arab";
      {
        "_digits";
        "٠١٢٣٤٥٦٧٨٩",
        "_type";
        "numeric";
      }
      "arabext";
      {
        "_digits";
        "٠١٢٣٤٥٦٧٨٩",
        "_type";
        "numeric";
      }
      "armn";
      {
        "_rules";
        "armenian-upper",
        "_type";
        "algorithmic";
      }
      "armnlow";
    }
  }
}

```

```

    {
      "_rules";
      "armenian-lower",
      "_type";
      "algorithmic";
    }
    "bali";
    {
      "_digits";
      "ᮀᮁᮂᮃᮄᮅᮆᮇᮈᮉ",
      "_type";
      "numeric";
    }
    "beng";
    {
      "_digits";
      "০১২৩৪৫৬৭৮৯",
      "_type";
      "numeric";
    }
    "bhks";
    {
      "_digits";
      "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉",
      "_type";
      "numeric";
    }
    "brah";
    {
      "_digits";
      "ᱠᱡᱣᱤᱦᱧᱨᱱᱲᱳ",
      "_type";
      "numeric";
    }
    "cakm";
    {
      "_digits";
      "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
      "_type";
      "numeric";
    }
    "cham";
    {
      "_digits";
      "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
      "_type";
      "numeric";
    }
    "cyr1";
    {
      "_rules";
      "cyrillic-lower",
      "_type";
      "algorithmic";
    }
    "deva";

```



```
{
  "_digits";
  "ፀፁፃፄፅፆፇፈ",
  "_type";
  "numeric";
}
"ethi";
{
  "_rules";
  "ethiopic",
  "_type";
  "algorithmic";
}
"fullwide";
{
  "_digits";
  "0 1 2 3 4 5 6 7 8 9",
  "_type";
  "numeric";
}
"geor";
{
  "_rules";
  "georgian",
  "_type";
  "algorithmic";
}
"grek";
{
  "_rules";
  "greek-upper",
  "_type";
  "algorithmic";
}
"greklow";
{
  "_rules";
  "greek-lower",
  "_type";
  "algorithmic";
}
"gujr";
{
  "_digits";
  "૦૧૨૩૪૫૬૭૮૯",
  "_type";
  "numeric";
}
"guru";
{
  "_digits";
  "੦੧੨੩੪੫੬੭੮੯",
  "_type";
  "numeric";
}
"hanidays";
```

```

    {
      "_rules";
      "zh/SpelloutRules/spellout-numbering-days",
      "_type";
      "algorithmic";
    }
    "hanidec";
    {
      "_digits";
      "〇一二三四五六七八九",
      "_type";
      "numeric";
    }
    "hans";
    {
      "_rules";
      "zh/SpelloutRules/spellout-cardinal",
      "_type";
      "algorithmic";
    }
    "hansfin";
    {
      "_rules";
      "zh/SpelloutRules/spellout-cardinal-financial",
      "_type";
      "algorithmic";
    }
    "hant";
    {
      "_rules";
      "zh_Hant/SpelloutRules/spellout-cardinal",
      "_type";
      "algorithmic";
    }
    "hantfin";
    {
      "_rules";
      "zh_Hant/SpelloutRules/spellout-cardinal-financial",
      "_type";
      "algorithmic";
    }
    "hebr";
    {
      "_rules";
      "hebrew",
      "_type";
      "algorithmic";
    }
    "hmng";
    {
      "_digits";
      "□□□□□□□□□□",
      "_type";
      "numeric";
    }
    "java";

```

```

    {
      "_digits";
      "၀၁၂၃၄၅၆၇၈၉၀၁၂၃၄၅၆၇၈၉",
      "_type";
      "numeric";
    }
    "jpan";
    {
      "_rules";
      "ja/SpelloutRules/spellout-cardinal",
      "_type";
      "algorithmic";
    }
    "jpanfin";
    {
      "_rules";
      "ja/SpelloutRules/spellout-cardinal-financial",
      "_type";
      "algorithmic";
    }
    "kali";
    {
      "_digits";
      "၀၀၀၀၀၀၀၀၀၀",
      "_type";
      "numeric";
    }
    "khmr";
    {
      "_digits";
      "០១២៣៤៥៦៧៨៩",
      "_type";
      "numeric";
    }
    "knda";
    {
      "_digits";
      "೦೧೨೩೪೫೬೭೮೯",
      "_type";
      "numeric";
    }
    "lana";
    {
      "_digits";
      "၀၀၀၀၀၀၀၀၀၀",
      "_type";
      "numeric";
    }
    "lanatham";
    {
      "_digits";
      "၀၀၀၀၀၀၀၀၀၀",
      "_type";
      "numeric";
    }
  }

```

```
"lao";
{
  "_digits";
  "໐໑໒໓໔໕໖໗໘໑",
  "_type";
  "numeric";
}
"latn";
{
  "_digits";
  "0123456789",
  "_type";
  "numeric";
}
"lepc";
{
  "_digits";
  "໐໑໒໓໔໕໖໗໘໑",
  "_type";
  "numeric";
}
"limb";
{
  "_digits";
  "໐໑໒໓໔໕໖໗໘໑",
  "_type";
  "numeric";
}
"mathbold";
{
  "_digits";
  "0123456789",
  "_type";
  "numeric";
}
"mathdbl";
{
  "_digits";
  "0123456789",
  "_type";
  "numeric";
}
"mathmono";
{
  "_digits";
  "0123456789",
  "_type";
  "numeric";
}
"mathsanb";
{
  "_digits";
  "0123456789",
  "_type";
  "numeric";
}
```

```

"mathsans";
{
  "_digits";
  "0123456789",
  "_type";
  "numeric";
}
"mlym";
{
  "_digits";
  "ഘറനൗറുനൗവുൗ",
  "_type";
  "numeric";
}
"modi";
{
  "_digits";
  "□□□□□□□□",
  "_type";
  "numeric";
}
"mong";
{
  "_digits";
  "ᠮᠣᠩᠭᠤᠯᠠ",
  "_type";
  "numeric";
}
"mroo";
{
  "_digits";
  "□□□□□□□□",
  "_type";
  "numeric";
}
"mtei";
{
  "_digits";
  "ᠮᠤᠳᠤᠢ",
  "_type";
  "numeric";
}
"mymr";
{
  "_digits";
  "ᠮᠤᠮᠣᠷ",
  "_type";
  "numeric";
}
"mymrshan";
{
  "_digits";
  "0123456789",
  "_type";
  "numeric";
}

```

```

}
"mymrtlng";
{
    "_digits";
    "0ᄇᄃ3ᄃ6ᄃᄃᄃᄃᄃᄃ",
    "_type";
    "numeric";
}
"newa";
{
    "_digits";
    "□□□□□□□□□□",
    "_type";
    "numeric";
}
"nkoo";
{
    "_digits";
    "ᄃᄃᄃᄃᄃᄃᄃᄃᄃᄃᄃᄃᄃ",
    "_type";
    "numeric";
}
"olck";
{
    "_digits";
    "0ᄃᄃᄃᄃᄃᄃᄃᄃᄃᄃᄃᄃᄃ",
    "_type";
    "numeric";
}
"orya";
{
    "_digits";
    "0ᄃᄃᄃᄃᄃᄃᄃᄃᄃᄃᄃᄃᄃ",
    "_type";
    "numeric";
}
"osma";
{
    "_digits";
    "0ᄃᄃᄃᄃᄃᄃᄃᄃᄃᄃᄃᄃᄃ",
    "_type";
    "numeric";
}
"roman";
{
    "_rules";
    "roman-upper",
    "_type";
    "algorithmic";
}
"romanlow";
{
    "_rules";
    "roman-lower",
    "type";
}

```

```

        "algorithmic";
    }
    "saur";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "shrd";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "sind";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "sinh";
    {
        "_digits";
        "ⁱᵀᵀᵀᵀᵀᵀᵀᵀᵀ",
        "_type";
        "numeric";
    }
    "sora";
    {
        "_digits";
        "ᵀᵀᵀᵀᵀᵀᵀᵀᵀ",
        "_type";
        "numeric";
    }
    "sund";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "takr";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "talv";
    {
        "_digits";
        "ᵀᵀᵀᵀᵀᵀᵀᵀᵀ",
        "_type";
    }

```

```

        "numeric";
    }
    "taml";
    {
        "_rules";
        "tamil",
        "_type";
        "algorithmic";
    }
    "tamldec";
    {
        "_digits";
        "௦௧௨௩௪௫௬௭௮௯",
        "_type";
        "numeric";
    }
    "telu";
    {
        "_digits";
        "౦౧౨౩౪౫౬౭౮౯",
        "_type";
        "numeric";
    }
    "thai";
    {
        "_digits";
        "๐๑๒๓๔๕๖๗๘๙",
        "_type";
        "numeric";
    }
    "tibet";
    {
        "_digits";
        "༠༡༢༣༤༥༦༧༨༩",
        "_type";
        "numeric";
    }
    "tirh";
    {
        "_digits";
        "୦୧୨୩୪୫୬୭୮୯",
        "_type";
        "numeric";
    }
    "vaih";
    {
        "_digits";
        "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍᱎᱏ᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱹᱺᱻᱼᱽ᱾᱿",
        "_type";
        "numeric";
    }
    "wara";
    {
        "_digits";
        "ᳵᳶ᳷᳸᳹ᳺ᳻᳼᳽᳾᳿",
        "_type";
    }

```



```

        "numeric";
    }
}
}
}

```

NUMBERS.JSON

```

{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31"
        },
        "language": "de"
      },
      "numbers": {
        "defaultNumberingSystem": "latn",
        "otherNumberingSystems": {
          "native": "latn"
        },
        "minimumGroupingDigits": "1",
        "symbols-numberSystem-latn": {
          "decimal": ",",
          "group": ".",
          "list": ";",
          "percentSign": "%",
          "plusSign": "+",
          "minusSign": "-",
          "exponential": "E",
          "superscriptingExponent": "·",
          "perMille": "‰",
          "infinity": "∞",
          "nan": "NaN",
          "timeSeparator": ":"
        },
        "decimalFormats-numberSystem-latn": {
          "standard": "#,##0.###",
          "long": {
            "decimalFormat": {
              "1000-count-one": "0 Tausend",
              "1000-count-other": "0 Tausend",
              "10000-count-one": "00 Tausend",
              "10000-count-other": "00 Tausend",
              "100000-count-one": "000 Tausend",
              "100000-count-other": "000 Tausend",
              "1000000-count-one": "0 Million",
              "1000000-count-other": "0 Millionen",
              "10000000-count-one": "00 Millionen",
              "10000000-count-other": "00 Millionen",
              "100000000-count-one": "000 Millionen",
              "100000000-count-other": "000 Millionen",
              "1000000000-count-one": "0 Milliarden",
              "1000000000-count-other": "0 Milliarden",
            }
          }
        }
      }
    }
  }
}

```

```

        "10000000000-count-one": "00 Milliarden",
        "10000000000-count-other": "00 Milliarden",
        "100000000000-count-one": "000 Milliarden",
        "100000000000-count-other": "000 Milliarden",
        "1000000000000-count-one": "0 Billion",
        "1000000000000-count-other": "0 Billionen",
        "10000000000000-count-one": "00 Billionen",
        "10000000000000-count-other": "00 Billionen",
        "100000000000000-count-one": "000 Billionen",
        "100000000000000-count-other": "000 Billionen"
    }
},
"short": {
    "decimalFormat": {
        "1000-count-one": "0",
        "1000-count-other": "0",
        "10000-count-one": "0",
        "10000-count-other": "0",
        "100000-count-one": "0",
        "100000-count-other": "0",
        "1000000-count-one": "0 Mio'.'",
        "1000000-count-other": "0 Mio'.'",
        "10000000-count-one": "00 Mio'.'",
        "10000000-count-other": "00 Mio'.'",
        "100000000-count-one": "000 Mio'.'",
        "100000000-count-other": "000 Mio'.'",
        "1000000000-count-one": "0 Mrd'.'",
        "1000000000-count-other": "0 Mrd'.'",
        "10000000000-count-one": "00 Mrd'.'",
        "10000000000-count-other": "00 Mrd'.'",
        "100000000000-count-one": "000 Mrd'.'",
        "100000000000-count-other": "000 Mrd'.'",
        "1000000000000-count-one": "0 Bio'.'",
        "1000000000000-count-other": "0 Bio'.'",
        "10000000000000-count-one": "00 Bio'.'",
        "10000000000000-count-other": "00 Bio'.'",
        "100000000000000-count-one": "000 Bio'.'",
        "100000000000000-count-other": "000 Bio'.'"
    }
}
},
"scientificFormats-numberSystem-latn": {
    "standard": "#E0"
},
"percentFormats-numberSystem-latn": {
    "standard": "#,##0 %"
},
"currencyFormats-numberSystem-latn": {
    "currencySpacing": {
        "beforeCurrency": {
            "currencyMatch": "[:^S:]",
            "surroundingMatch": "[:digit:]",
            "insertBetween": " "
        },
        "afterCurrency": {
            "currencyMatch": "[:^S:]",
            "surroundingMatch": "[:digit:]",

```

```

    "insertBetween": " "
  },
  "standard": "#,##0.00 ¤",
  "accounting": "#,##0.00 ¤",
  "short": {
    "standard": {
      "1000-count-one": "0 Tsd'.' ¤",
      "1000-count-other": "0 Tsd'.' ¤",
      "10000-count-one": "00 Tsd'.' ¤",
      "10000-count-other": "00 Tsd'.' ¤",
      "100000-count-one": "000 Tsd'.' ¤",
      "100000-count-other": "000 Tsd'.' ¤",
      "1000000-count-one": "0 Mio'.' ¤",
      "1000000-count-other": "0 Mio'.' ¤",
      "10000000-count-one": "00 Mio'.' ¤",
      "10000000-count-other": "00 Mio'.' ¤",
      "100000000-count-one": "000 Mio'.' ¤",
      "100000000-count-other": "000 Mio'.' ¤",
      "1000000000-count-one": "0 Mrd'.' ¤",
      "1000000000-count-other": "0 Mrd'.' ¤",
      "10000000000-count-one": "00 Mrd'.' ¤",
      "10000000000-count-other": "00 Mrd'.' ¤",
      "100000000000-count-one": "000 Mrd'.' ¤",
      "100000000000-count-other": "000 Mrd'.' ¤",
      "1000000000000-count-one": "0 Bio'.' ¤",
      "1000000000000-count-other": "0 Bio'.' ¤",
      "10000000000000-count-one": "00 Bio'.' ¤",
      "10000000000000-count-other": "00 Bio'.' ¤",
      "100000000000000-count-one": "000 Bio'.' ¤",
      "100000000000000-count-other": "000 Bio'.' ¤"
    }
  },
  "unitPattern-count-one": "{0} {1}",
  "unitPattern-count-other": "{0} {1}"
},
"miscPatterns-numberSystem-latn": {
  "atLeast": "{0}+",
  "range": "{0}-{1}"
},
"minimalPairs": {
  "pluralMinimalPairs": "{0} Tag",
  "pluralMinimalPairs": "{0} Tage",
  "other": "{0}. Abzweigung nach rechts nehmen"
}
}
}
}
}

```

NUMBERS.JSX

```
{
    "main";
    {
        "de";
    }
}
```

```

{
  "identity";
  {
    "version";
    {
      "_number";
      "$Revision: 13259 $",
      "_cldrVersion";
      "31";
    }
    "language";
    "de";
  }
  "numbers";
  {
    "defaultNumberingSystem";
    "latn",
    "otherNumberingSystems";
    {
      "native";
      "latn";
    }
    "minimumGroupingDigits";
    "1",
    "symbols-numberSystem-latn";
    {
      "decimal";
      ",",
      "group";
      ".",
      "list";
      ";",
      "percentSign";
      "%",
      "plusSign";
      "+",
      "minusSign";
      "-",
      "exponential";
      "E",
      "superscriptingExponent";
      ". ",
      "perMille";
      "‰",
      "infinity";
      "∞",
      "nan";
      "NaN",
      "timeSeparator";
      ":";
    }
    "decimalFormats-numberSystem-latn";
    {
      "standard";
      "#,##0.###",
      "long";
    }
  }
}

```

```

        "decimalFormat";
        {
            "1000-count-one";
            "0 Tausend",
                "1000-count-other";
            "0 Tausend",
                "10000-count-one";
            "00 Tausend",
                "10000-count-other";
            "00 Tausend",
                "100000-count-one";
            "000 Tausend",
                "100000-count-other";
            "000 Tausend",
                "1000000-count-one";
            "0 Million",
                "1000000-count-other";
            "0 Millionen",
                "10000000-count-one";
            "00 Millionen",
                "10000000-count-other";
            "00 Millionen",
                "100000000-count-one";
            "000 Millionen",
                "100000000-count-other";
            "000 Millionen",
                "1000000000-count-one";
            "0 Milliarde",
                "1000000000-count-other";
            "0 Milliarden",
                "10000000000-count-one";
            "00 Milliarden",
                "10000000000-count-other";
            "00 Milliarden",
                "100000000000-count-one";
            "000 Milliarden",
                "100000000000-count-other";
            "000 Milliarden",
                "1000000000000-count-one";
            "0 Billion",
                "1000000000000-count-other";
            "0 Billionen",
                "10000000000000-count-one";
            "00 Billionen",
                "10000000000000-count-other";
            "00 Billionen",
                "100000000000000-count-one";
            "000 Billionen",
                "100000000000000-count-other";
            "000 Billionen";
        }
    }
    "short";
    {
        "decimalFormat";
        {
            "1000-count-one";

```

```

        "0",
        "1000-count-other";
        "0",
        "10000-count-one";
        "0",
        "10000-count-other";
        "0",
        "100000-count-one";
        "0",
        "100000-count-other";
        "0",
        "1000000-count-one";
        "0 Mio'.'",
        "1000000-count-other";
        "0 Mio'.'",
        "10000000-count-one";
        "00 Mio'.'",
        "10000000-count-other";
        "00 Mio'.'",
        "100000000-count-one";
        "000 Mio'.'",
        "100000000-count-other";
        "000 Mio'.'",
        "1000000000-count-one";
        "0 Mrd'.'",
        "1000000000-count-other";
        "0 Mrd'.'",
        "10000000000-count-one";
        "00 Mrd'.'",
        "10000000000-count-other";
        "00 Mrd'.'",
        "100000000000-count-one";
        "000 Mrd'.'",
        "100000000000-count-other";
        "000 Mrd'.'",
        "1000000000000-count-one";
        "0 Bio'.'",
        "1000000000000-count-other";
        "0 Bio'.'",
        "10000000000000-count-one";
        "00 Bio'.'",
        "10000000000000-count-other";
        "00 Bio'.'",
        "100000000000000-count-one";
        "000 Bio'.'",
        "100000000000000-count-other";
        "000 Bio'.'";
    }
}
}
"scientificFormats-numberSystem-latn";
{
    "standard";
    "#E0";
}
"percentFormats-numberSystem-latn";
{

```

```

        "standard";
        "#,##0 %";
    }
    "currencyFormats-numberSystem-latn";
    {
        "currencySpacing";
        {
            "beforeCurrency";
            {
                "currencyMatch";
                "[:^S:]",
                "surroundingMatch";
                "[:digit:]",
                "insertBetween";
                " ";
            }
            "afterCurrency";
            {
                "currencyMatch";
                "[:^S:]",
                "surroundingMatch";
                "[:digit:]",
                "insertBetween";
                " ";
            }
        }
    }
    "standard";
    "#,##0.00 ¤",
    "accounting";
    "#,##0.00 ¤",
    "short";
    {
        "standard";
        {
            "1000-count-one";
            "0 Tsd'.' ¤",
            "1000-count-other";
            "0 Tsd'.' ¤",
            "10000-count-one";
            "00 Tsd'.' ¤",
            "10000-count-other";
            "00 Tsd'.' ¤",
            "100000-count-one";
            "000 Tsd'.' ¤",
            "100000-count-other";
            "000 Tsd'.' ¤",
            "1000000-count-one";
            "0 Mio'.' ¤",
            "1000000-count-other";
            "0 Mio'.' ¤",
            "10000000-count-one";
            "00 Mio'.' ¤",
            "10000000-count-other";
            "00 Mio'.' ¤",
            "100000000-count-one";
            "000 Mio'.' ¤",
            "100000000-count-other";
        }
    }

```

```

        "000 Mio'.' s",
        "1000000000-count-one";
    "0 Mrd'.' s",
        "1000000000-count-other";
    "0 Mrd'.' s",
        "1000000000-count-one";
    "00 Mrd'.' s",
        "1000000000-count-other";
    "00 Mrd'.' s",
        "10000000000-count-one";
    "000 Mrd'.' s",
        "10000000000-count-other";
    "000 Mrd'.' s",
        "100000000000-count-one";
    "0 Bio'.' s",
        "100000000000-count-other";
    "0 Bio'.' s",
        "1000000000000-count-one";
    "00 Bio'.' s",
        "1000000000000-count-other";
    "00 Bio'.' s",
        "10000000000000-count-one";
    "000 Bio'.' s",
        "10000000000000-count-other";
    "000 Bio'.' s";
}
}
"unitPattern-count-one";
"{0} {1}",
    "unitPattern-count-other";
"{0} {1}";
}
"miscPatterns-numberSystem-latn";
{
    "atLeast";
    "{0}+",
        "range";
    "{0}-{1}";
}
"minimalPairs";
{
    "pluralMinimalPairs";
    "{0} Tag",
        "pluralMinimalPairs";
    "{0} Tage",
        "other";
    "{0}. Abzweigung nach rechts nehmen";
}
}
}
}

```

TIMEZONENAMES.JSON

$$\{$$


```
"main": {
  "de": {
    "identity": {
      "version": {
        "_number": "$Revision: 12879 $",
        "_cldrVersion": "30.0.3"
      },
      "language": "de"
    },
    "dates": {
      "timeZoneNames": {
        "hourFormat": "+HH:mm;-HH:mm",
        "gmtFormat": "GMT{0}",
        "gmtZeroFormat": "GMT",
        "regionFormat": "{0} Zeit",
        "regionFormat-type-daylight": "{0} Sommerzeit",
        "regionFormat-type-standard": "{0} Normalzeit",
        "fallbackFormat": "{1} ({0})",
        "zone": {
          "America": {
            "Adak": {
              "exemplarCity": "Adak"
            },
            "Anchorage": {
              "exemplarCity": "Anchorage"
            },
            "Anguilla": {
              "exemplarCity": "Anguilla"
            },
            "Antigua": {
              "exemplarCity": "Antigua"
            },
            "Araguaina": {
              "exemplarCity": "Araguaina"
            },
            "Argentina": {
              "Rio_Gallegos": {
                "exemplarCity": "Rio Gallegos"
              },
              "San_Juan": {
                "exemplarCity": "San Juan"
              },
              "Ushuaia": {
                "exemplarCity": "Ushuaia"
              },
              "La_Rioja": {
                "exemplarCity": "La Rioja"
              },
              "San_Luis": {
                "exemplarCity": "San Luis"
              },
              "Salta": {
                "exemplarCity": "Salta"
              },
              "Tucuman": {
                "exemplarCity": "Tucuman"
              }
            }
          }
        }
      }
    }
  }
}
```

```
    },  
    "Aruba": {  
      "exemplarCity": "Aruba"  
    },  
    "Asuncion": {  
      "exemplarCity": "Asunción"  
    },  
    "Bahia": {  
      "exemplarCity": "Bahia"  
    },  
    "Bahia_Banderas": {  
      "exemplarCity": "Bahia Banderas"  
    },  
    "Barbados": {  
      "exemplarCity": "Barbados"  
    },  
    "Belem": {  
      "exemplarCity": "Belem"  
    },  
    "Belize": {  
      "exemplarCity": "Belize"  
    },  
    "Blanc-Sablon": {  
      "exemplarCity": "Blanc-Sablon"  
    },  
    "Boa_Vista": {  
      "exemplarCity": "Boa Vista"  
    },  
    "Bogota": {  
      "exemplarCity": "Bogotá"  
    },  
    "Boise": {  
      "exemplarCity": "Boise"  
    },  
    "Buenos_Aires": {  
      "exemplarCity": "Buenos Aires"  
    },  
    "Cambridge_Bay": {  
      "exemplarCity": "Cambridge Bay"  
    },  
    "Campo_Grande": {  
      "exemplarCity": "Campo Grande"  
    },  
    "Cancun": {  
      "exemplarCity": "Cancún"  
    },  
    "Caracas": {  
      "exemplarCity": "Caracas"  
    },  
    "Catamarca": {  
      "exemplarCity": "Catamarca"  
    },  
    "Cayenne": {  
      "exemplarCity": "Cayenne"  
    },  
    "Cayman": {  
      "exemplarCity": "Kaimaninseln"
```

```
    },
    "Chicago": {
      "exemplarCity": "Chicago"
    },
    "Chihuahua": {
      "exemplarCity": "Chihuahua"
    },
    "Coral_Harbour": {
      "exemplarCity": "Atikokan"
    },
    "Cordoba": {
      "exemplarCity": "Córdoba"
    },
    "Costa_Rica": {
      "exemplarCity": "Costa Rica"
    },
    "Creston": {
      "exemplarCity": "Creston"
    },
    "Cuiaba": {
      "exemplarCity": "Cuiaba"
    },
    "Curacao": {
      "exemplarCity": "Curaçao"
    },
    "Danmarkshavn": {
      "exemplarCity": "Danmarkshavn"
    },
    "Dawson": {
      "exemplarCity": "Dawson"
    },
    "Dawson_Creek": {
      "exemplarCity": "Dawson Creek"
    },
    "Denver": {
      "exemplarCity": "Denver"
    },
    "Detroit": {
      "exemplarCity": "Detroit"
    },
    "Dominica": {
      "exemplarCity": "Dominica"
    },
    "Edmonton": {
      "exemplarCity": "Edmonton"
    },
    "Eirunepe": {
      "exemplarCity": "Eirunepe"
    },
    "El_Salvador": {
      "exemplarCity": "El Salvador"
    },
    "Fort_Nelson": {
      "exemplarCity": "Fort Nelson"
    },
    "Fortaleza": {
      "exemplarCity": "Fortaleza"
    }
  },
  "exemplarCity": "Fortaleza"
}
```

```
    },
    "Glace_Bay": {
      "exemplarCity": "Glace Bay"
    },
    "Godthab": {
      "exemplarCity": "Nuuk"
    },
    "Goose_Bay": {
      "exemplarCity": "Goose Bay"
    },
    "Grand_Turk": {
      "exemplarCity": "Grand Turk"
    },
    "Grenada": {
      "exemplarCity": "Grenada"
    },
    "Guadeloupe": {
      "exemplarCity": "Guadeloupe"
    },
    "Guatemala": {
      "exemplarCity": "Guatemala"
    },
    "Guayaquil": {
      "exemplarCity": "Guayaquil"
    },
    "Guyana": {
      "exemplarCity": "Guyana"
    },
    "Halifax": {
      "exemplarCity": "Halifax"
    },
    "Havana": {
      "exemplarCity": "Havanna"
    },
    "Hermosillo": {
      "exemplarCity": "Hermosillo"
    },
    "Indiana": {
      "Vincennes": {
        "exemplarCity": "Vincennes, Indiana"
      },
      "Petersburg": {
        "exemplarCity": "Petersburg, Indiana"
      },
      "Tell_City": {
        "exemplarCity": "Tell City, Indiana"
      },
      "Knox": {
        "exemplarCity": "Knox, Indiana"
      },
      "Winamac": {
        "exemplarCity": "Winamac, Indiana"
      },
      "Marengo": {
        "exemplarCity": "Marengo, Indiana"
      },
      "Vevay": {
```

```
        "exemplarCity": "Vevay, Indiana"
      },
      "Indianapolis": {
        "exemplarCity": "Indianapolis"
      },
      "Inuvik": {
        "exemplarCity": "Inuvik"
      },
      "Iqaluit": {
        "exemplarCity": "Iqaluit"
      },
      "Jamaica": {
        "exemplarCity": "Jamaika"
      },
      "Jujuy": {
        "exemplarCity": "Jujuy"
      },
      "Juneau": {
        "exemplarCity": "Juneau"
      },
      "Kentucky": {
        "Monticello": {
          "exemplarCity": "Monticello, Kentucky"
        }
      },
      "Kralendijk": {
        "exemplarCity": "Kralendijk"
      },
      "La_Paz": {
        "exemplarCity": "La Paz"
      },
      "Lima": {
        "exemplarCity": "Lima"
      },
      "Los_Angeles": {
        "exemplarCity": "Los Angeles"
      },
      "Louisville": {
        "exemplarCity": "Louisville"
      },
      "Lower_Princes": {
        "exemplarCity": "Lower Prince's Quarter"
      },
      "Maceio": {
        "exemplarCity": "Maceio"
      },
      "Managua": {
        "exemplarCity": "Managua"
      },
      "Manaus": {
        "exemplarCity": "Manaus"
      },
      "Marigot": {
        "exemplarCity": "Marigot"
      },
      "Martinique": {
```

```
    "exemplarCity": "Martinique"
  },
  "Matamoros": {
    "exemplarCity": "Matamoros"
  },
  "Mazatlan": {
    "exemplarCity": "Mazatlan"
  },
  "Mendoza": {
    "exemplarCity": "Mendoza"
  },
  "Menominee": {
    "exemplarCity": "Menominee"
  },
  "Merida": {
    "exemplarCity": "Merida"
  },
  "Metlakatla": {
    "exemplarCity": "Metlakatla"
  },
  "Mexico_City": {
    "exemplarCity": "Mexiko-Stadt"
  },
  "Miquelon": {
    "exemplarCity": "Miquelon"
  },
  "Moncton": {
    "exemplarCity": "Moncton"
  },
  "Monterrey": {
    "exemplarCity": "Monterrey"
  },
  "Montevideo": {
    "exemplarCity": "Montevideo"
  },
  "Montserrat": {
    "exemplarCity": "Montserrat"
  },
  "Nassau": {
    "exemplarCity": "Nassau"
  },
  "New_York": {
    "exemplarCity": "New York"
  },
  "Nipigon": {
    "exemplarCity": "Nipigon"
  },
  "Nome": {
    "exemplarCity": "Nome"
  },
  "Noronha": {
    "exemplarCity": "Noronha"
  },
  "North_Dakota": {
    "Beulah": {
      "exemplarCity": "Beulah, North Dakota"
    },
  },
```

```
    "New_Salem": {
      "exemplarCity": "New Salem, North Dakota"
    },
    "Center": {
      "exemplarCity": "Center, North Dakota"
    }
  },
  "Ojinaga": {
    "exemplarCity": "Ojinaga"
  },
  "Panama": {
    "exemplarCity": "Panama"
  },
  "Pangnirtung": {
    "exemplarCity": "Pangnirtung"
  },
  "Paramaribo": {
    "exemplarCity": "Paramaribo"
  },
  "Phoenix": {
    "exemplarCity": "Phoenix"
  },
  "Port-au-Prince": {
    "exemplarCity": "Port-au-Prince"
  },
  "Port_of_Spain": {
    "exemplarCity": "Port of Spain"
  },
  "Porto_Velho": {
    "exemplarCity": "Porto Velho"
  },
  "Puerto_Rico": {
    "exemplarCity": "Puerto Rico"
  },
  "Rainy_River": {
    "exemplarCity": "Rainy River"
  },
  "Rankin_Inlet": {
    "exemplarCity": "Rankin Inlet"
  },
  "Recife": {
    "exemplarCity": "Recife"
  },
  "Regina": {
    "exemplarCity": "Regina"
  },
  "Resolute": {
    "exemplarCity": "Resolute"
  },
  "Rio_Branco": {
    "exemplarCity": "Rio Branco"
  },
  "Santa_Isabel": {
    "exemplarCity": "Santa Isabel"
  },
  "Santarem": {
    "exemplarCity": "Santarem"
  }
```

```
    },
    "Santiago": {
      "exemplarCity": "Santiago"
    },
    "Santo_Domingo": {
      "exemplarCity": "Santo Domingo"
    },
    "Sao_Paulo": {
      "exemplarCity": "São Paulo"
    },
    "Scoresbysund": {
      "exemplarCity": "Ittoqqortoormiit"
    },
    "Sitka": {
      "exemplarCity": "Sitka"
    },
    "St_Barthelemy": {
      "exemplarCity": "Saint-Barthélemy"
    },
    "St_Johns": {
      "exemplarCity": "St. John's"
    },
    "St_Kitts": {
      "exemplarCity": "St. Kitts"
    },
    "St_Lucia": {
      "exemplarCity": "St. Lucia"
    },
    "St_Thomas": {
      "exemplarCity": "St. Thomas"
    },
    "St_Vincent": {
      "exemplarCity": "St. Vincent"
    },
    "Swift_Current": {
      "exemplarCity": "Swift Current"
    },
    "Tegucigalpa": {
      "exemplarCity": "Tegucigalpa"
    },
    "Thule": {
      "exemplarCity": "Thule"
    },
    "Thunder_Bay": {
      "exemplarCity": "Thunder Bay"
    },
    "Tijuana": {
      "exemplarCity": "Tijuana"
    },
    "Toronto": {
      "exemplarCity": "Toronto"
    },
    "Tortola": {
      "exemplarCity": "Tortola"
    },
    "Vancouver": {
      "exemplarCity": "Vancouver"
    }
  }
}
```



```
    },
    "Whitehorse": {
      "exemplarCity": "Whitehorse"
    },
    "Winnipeg": {
      "exemplarCity": "Winnipeg"
    },
    "Yakutat": {
      "exemplarCity": "Yakutat"
    },
    "Yellowknife": {
      "exemplarCity": "Yellowknife"
    }
  },
  "Atlantic": {
    "Azores": {
      "exemplarCity": "Azoren"
    },
    "Bermuda": {
      "exemplarCity": "Bermudas"
    },
    "Canary": {
      "exemplarCity": "Kanaren"
    },
    "Cape_Verde": {
      "exemplarCity": "Cabo Verde"
    },
    "Faeroe": {
      "exemplarCity": "Färöer"
    },
    "Madeira": {
      "exemplarCity": "Madeira"
    },
    "Reykjavik": {
      "exemplarCity": "Reykjavík"
    },
    "South_Georgia": {
      "exemplarCity": "Südgeorgien"
    },
    "St_Helena": {
      "exemplarCity": "St. Helena"
    },
    "Stanley": {
      "exemplarCity": "Stanley"
    }
  },
  "Europe": {
    "Amsterdam": {
      "exemplarCity": "Amsterdam"
    },
    "Andorra": {
      "exemplarCity": "Andorra"
    },
    "Astrakhan": {
      "exemplarCity": "Astrachan"
    },
    "Athens": {
```

```
    "exemplarCity": "Athen"
  },
  "Belgrade": {
    "exemplarCity": "Belgrad"
  },
  "Berlin": {
    "exemplarCity": "Berlin"
  },
  "Bratislava": {
    "exemplarCity": "Bratislava"
  },
  "Brussels": {
    "exemplarCity": "Brüssel"
  },
  "Bucharest": {
    "exemplarCity": "Bukarest"
  },
  "Budapest": {
    "exemplarCity": "Budapest"
  },
  "Busingen": {
    "exemplarCity": "Büsingen"
  },
  "Chisinau": {
    "exemplarCity": "Kischinau"
  },
  "Copenhagen": {
    "exemplarCity": "Kopenhagen"
  },
  "Dublin": {
    "long": {
      "daylight": "Irische Sommerzeit"
    },
    "exemplarCity": "Dublin"
  },
  "Gibraltar": {
    "exemplarCity": "Gibraltar"
  },
  "Guernsey": {
    "exemplarCity": "Guernsey"
  },
  "Helsinki": {
    "exemplarCity": "Helsinki"
  },
  "Isle_of_Man": {
    "exemplarCity": "Isle of Man"
  },
  "Istanbul": {
    "exemplarCity": "Istanbul"
  },
  "Jersey": {
    "exemplarCity": "Jersey"
  },
  "Kaliningrad": {
    "exemplarCity": "Kaliningrad"
  },
  "Kiev": {
```

```
        "exemplarCity": "Kiew"
      },
      "Kirov": {
        "exemplarCity": "Kirow"
      },
      "Lisbon": {
        "exemplarCity": "Lissabon"
      },
      "Ljubljana": {
        "exemplarCity": "Ljubljana"
      },
      "London": {
        "long": {
          "daylight": "Britische Sommerzeit"
        },
        "exemplarCity": "London"
      },
      "Luxembourg": {
        "exemplarCity": "Luxemburg"
      },
      "Madrid": {
        "exemplarCity": "Madrid"
      },
      "Malta": {
        "exemplarCity": "Malta"
      },
      "Mariehamn": {
        "exemplarCity": "Mariehamn"
      },
      "Minsk": {
        "exemplarCity": "Minsk"
      },
      "Monaco": {
        "exemplarCity": "Monaco"
      },
      "Moscow": {
        "exemplarCity": "Moskau"
      },
      "Oslo": {
        "exemplarCity": "Oslo"
      },
      "Paris": {
        "exemplarCity": "Paris"
      },
      "Podgorica": {
        "exemplarCity": "Podgorica"
      },
      "Prague": {
        "exemplarCity": "Prag"
      },
      "Riga": {
        "exemplarCity": "Riga"
      },
      "Rome": {
        "exemplarCity": "Rom"
      },
      "Samara": {
```

```
    "exemplarCity": "Samara"
  },
  "San_Marino": {
    "exemplarCity": "San Marino"
  },
  "Sarajevo": {
    "exemplarCity": "Sarajevo"
  },
  "Simferopol": {
    "exemplarCity": "Simferopol"
  },
  "Skopje": {
    "exemplarCity": "Skopje"
  },
  "Sofia": {
    "exemplarCity": "Sofia"
  },
  "Stockholm": {
    "exemplarCity": "Stockholm"
  },
  "Tallinn": {
    "exemplarCity": "Tallinn"
  },
  "Tirane": {
    "exemplarCity": "Tirana"
  },
  "Ulyanovsk": {
    "exemplarCity": "Uljanowsk"
  },
  "Uzhgorod": {
    "exemplarCity": "Uschgorod"
  },
  "Vaduz": {
    "exemplarCity": "Vaduz"
  },
  "Vatican": {
    "exemplarCity": "Vatikan"
  },
  "Vienna": {
    "exemplarCity": "Wien"
  },
  "Vilnius": {
    "exemplarCity": "Vilnius"
  },
  "Volgograd": {
    "exemplarCity": "Wolgograd"
  },
  "Warsaw": {
    "exemplarCity": "Warschau"
  },
  "Zagreb": {
    "exemplarCity": "Zagreb"
  },
  "Zaporozhye": {
    "exemplarCity": "Saporischja"
  },
  "Zurich": {
```

```
        "exemplarCity": "Zürich"
      },
    },
    "Africa": {
      "Abidjan": {
        "exemplarCity": "Abidjan"
      },
      "Accra": {
        "exemplarCity": "Accra"
      },
      "Addis_Ababa": {
        "exemplarCity": "Addis Abeba"
      },
      "Algiers": {
        "exemplarCity": "Algier"
      },
      "Asmera": {
        "exemplarCity": "Asmara"
      },
      "Bamako": {
        "exemplarCity": "Bamako"
      },
      "Bangui": {
        "exemplarCity": "Bangui"
      },
      "Banjul": {
        "exemplarCity": "Banjul"
      },
      "Bissau": {
        "exemplarCity": "Bissau"
      },
      "Blantyre": {
        "exemplarCity": "Blantyre"
      },
      "Brazzaville": {
        "exemplarCity": "Brazzaville"
      },
      "Bujumbura": {
        "exemplarCity": "Bujumbura"
      },
      "Cairo": {
        "exemplarCity": "Kairo"
      },
      "Casablanca": {
        "exemplarCity": "Casablanca"
      },
      "Ceuta": {
        "exemplarCity": "Ceuta"
      },
      "Conakry": {
        "exemplarCity": "Conakry"
      },
      "Dakar": {
        "exemplarCity": "Dakar"
      },
      "Dar_es_Salaam": {
        "exemplarCity": "Daressalam"
      }
    }
  }
}
```

```
    },  
    "Djibouti": {  
      "exemplarCity": "Dschibuti"  
    },  
    "Douala": {  
      "exemplarCity": "Douala"  
    },  
    "El_Aaiun": {  
      "exemplarCity": "El Aaiún"  
    },  
    "Freetown": {  
      "exemplarCity": "Freetown"  
    },  
    "Gaborone": {  
      "exemplarCity": "Gaborone"  
    },  
    "Harare": {  
      "exemplarCity": "Harare"  
    },  
    "Johannesburg": {  
      "exemplarCity": "Johannesburg"  
    },  
    "Juba": {  
      "exemplarCity": "Juba"  
    },  
    "Kampala": {  
      "exemplarCity": "Kampala"  
    },  
    "Khartoum": {  
      "exemplarCity": "Khartum"  
    },  
    "Kigali": {  
      "exemplarCity": "Kigali"  
    },  
    "Kinshasa": {  
      "exemplarCity": "Kinshasa"  
    },  
    "Lagos": {  
      "exemplarCity": "Lagos"  
    },  
    "Libreville": {  
      "exemplarCity": "Libreville"  
    },  
    "Lome": {  
      "exemplarCity": "Lomé"  
    },  
    "Luanda": {  
      "exemplarCity": "Luanda"  
    },  
    "Lubumbashi": {  
      "exemplarCity": "Lubumbashi"  
    },  
    "Lusaka": {  
      "exemplarCity": "Lusaka"  
    },  
    "Malabo": {  
      "exemplarCity": "Malabo"
```

```
    },  
    "Maputo": {  
      "exemplarCity": "Maputo"  
    },  
    "Maseru": {  
      "exemplarCity": "Maseru"  
    },  
    "Mbabane": {  
      "exemplarCity": "Mbabane"  
    },  
    "Mogadishu": {  
      "exemplarCity": "Mogadischu"  
    },  
    "Monrovia": {  
      "exemplarCity": "Monrovia"  
    },  
    "Nairobi": {  
      "exemplarCity": "Nairobi"  
    },  
    "Ndjamena": {  
      "exemplarCity": "N'Djamena"  
    },  
    "Niamey": {  
      "exemplarCity": "Niamey"  
    },  
    "Nouakchott": {  
      "exemplarCity": "Nouakchott"  
    },  
    "Ouagadougou": {  
      "exemplarCity": "Ouagadougou"  
    },  
    "Porto-Novo": {  
      "exemplarCity": "Porto Novo"  
    },  
    "Sao_Tome": {  
      "exemplarCity": "São Tomé"  
    },  
    "Tripoli": {  
      "exemplarCity": "Tripolis"  
    },  
    "Tunis": {  
      "exemplarCity": "Tunis"  
    },  
    "Windhoek": {  
      "exemplarCity": "Windhoek"  
    }  
  },  
  "Asia": {  
    "Aden": {  
      "exemplarCity": "Aden"  
    },  
    "Almaty": {  
      "exemplarCity": "Almaty"  
    },  
    "Amman": {  
      "exemplarCity": "Amman"  
    }  
  },  
}
```

```
"Anadyr": {
  "exemplarCity": "Anadyr"
},
"Aqtau": {
  "exemplarCity": "Aqtau"
},
"Aqtobe": {
  "exemplarCity": "Aktobe"
},
"Ashgabat": {
  "exemplarCity": "Aşgabat"
},
"Baghdad": {
  "exemplarCity": "Bagdad"
},
"Bahrain": {
  "exemplarCity": "Bahrain"
},
"Baku": {
  "exemplarCity": "Baku"
},
"Bangkok": {
  "exemplarCity": "Bangkok"
},
"Barnaul": {
  "exemplarCity": "Barnaul"
},
"Beirut": {
  "exemplarCity": "Beirut"
},
"Bishkek": {
  "exemplarCity": "Bischkek"
},
"Brunei": {
  "exemplarCity": "Brunei"
},
"Calcutta": {
  "exemplarCity": "Kalkutta"
},
"Chita": {
  "exemplarCity": "Tschita"
},
"Choibalsan": {
  "exemplarCity": "Tschoibalsan"
},
"Colombo": {
  "exemplarCity": "Colombo"
},
"Damascus": {
  "exemplarCity": "Damaskus"
},
"Dhaka": {
  "exemplarCity": "Dhaka"
},
"Dili": {
  "exemplarCity": "Dili"
},
},
```



```
"Dubai": {
  "exemplarCity": "Dubai"
},
"Dushanbe": {
  "exemplarCity": "Duschanbe"
},
"Gaza": {
  "exemplarCity": "Gaza"
},
"Hebron": {
  "exemplarCity": "Hebron"
},
"Hong_Kong": {
  "exemplarCity": "Hongkong"
},
"Hovd": {
  "exemplarCity": "Chowd"
},
"Irkutsk": {
  "exemplarCity": "Irkutsk"
},
"Jakarta": {
  "exemplarCity": "Jakarta"
},
"Jayapura": {
  "exemplarCity": "Jayapura"
},
"Jerusalem": {
  "exemplarCity": "Jerusalem"
},
"Kabul": {
  "exemplarCity": "Kabul"
},
"Kamchatka": {
  "exemplarCity": "Kamtschatka"
},
"Karachi": {
  "exemplarCity": "Karatschi"
},
"Katmandu": {
  "exemplarCity": "Kathmandu"
},
"Khandyga": {
  "exemplarCity": "Chandyga"
},
"Krasnoyarsk": {
  "exemplarCity": "Krasnojarsk"
},
"Kuala_Lumpur": {
  "exemplarCity": "Kuala Lumpur"
},
"Kuching": {
  "exemplarCity": "Kuching"
},
"Kuwait": {
  "exemplarCity": "Kuwait"
},
}
```

```
"Macau": {
  "exemplarCity": "Macao"
},
"Magadan": {
  "exemplarCity": "Magadan"
},
"Makassar": {
  "exemplarCity": "Makassar"
},
"Manila": {
  "exemplarCity": "Manila"
},
"Muscat": {
  "exemplarCity": "Maskat"
},
"Nicosia": {
  "exemplarCity": "Nikosia"
},
"Novokuznetsk": {
  "exemplarCity": "Nowokuznetsk"
},
"Novosibirsk": {
  "exemplarCity": "Nowosibirsk"
},
"Omsk": {
  "exemplarCity": "Omsk"
},
"Oral": {
  "exemplarCity": "Oral"
},
"Phnom_Penh": {
  "exemplarCity": "Phnom Penh"
},
"Pontianak": {
  "exemplarCity": "Pontianak"
},
"Pyongyang": {
  "exemplarCity": "Pjöngjang"
},
"Qatar": {
  "exemplarCity": "Katar"
},
"Qyzylorda": {
  "exemplarCity": "Qysylorda"
},
"Rangoon": {
  "exemplarCity": "Rangun"
},
"Riyadh": {
  "exemplarCity": "Riad"
},
"Saigon": {
  "exemplarCity": "Ho-Chi-Minh-Stadt"
},
"Sakhalin": {
  "exemplarCity": "Sachalin"
},
}
```

```
"Samarkand": {
  "exemplarCity": "Samarkand"
},
"Seoul": {
  "exemplarCity": "Seoul"
},
"Shanghai": {
  "exemplarCity": "Shanghai"
},
"Singapore": {
  "exemplarCity": "Singapur"
},
"Srednekolymsk": {
  "exemplarCity": "Srednekolymsk"
},
"Taipei": {
  "exemplarCity": "Taipeh"
},
"Tashkent": {
  "exemplarCity": "Taschkent"
},
"Tbilisi": {
  "exemplarCity": "Tiflis"
},
"Tehran": {
  "exemplarCity": "Teheran"
},
"Thimphu": {
  "exemplarCity": "Thimphu"
},
"Tokyo": {
  "exemplarCity": "Tokio"
},
"Tomsk": {
  "exemplarCity": "Tomsk"
},
"Ulaanbaatar": {
  "exemplarCity": "Ulaanbaatar"
},
"Urumqi": {
  "exemplarCity": "Ürümqi"
},
"Ust-Nera": {
  "exemplarCity": "Ust-Nera"
},
"Vientiane": {
  "exemplarCity": "Vientiane"
},
"Vladivostok": {
  "exemplarCity": "Wladiwostok"
},
"Yakutsk": {
  "exemplarCity": "Jakutsk"
},
"Yekaterinburg": {
  "exemplarCity": "Jekaterinburg"
},
}
```

```
    "Yerevan": {
      "exemplarCity": "Eriwan"
    },
    "Indian": {
      "Antananarivo": {
        "exemplarCity": "Antananarivo"
      },
      "Chagos": {
        "exemplarCity": "Chagos"
      },
      "Christmas": {
        "exemplarCity": "Weihnachtsinsel"
      },
      "Cocos": {
        "exemplarCity": "Cocos"
      },
      "Comoro": {
        "exemplarCity": "Komoren"
      },
      "Kerguelen": {
        "exemplarCity": "Kerguelen"
      },
      "Mahe": {
        "exemplarCity": "Mahe"
      },
      "Maldives": {
        "exemplarCity": "Malediven"
      },
      "Mauritius": {
        "exemplarCity": "Mauritius"
      },
      "Mayotte": {
        "exemplarCity": "Mayotte"
      },
      "Reunion": {
        "exemplarCity": "Réunion"
      }
    },
    "Australia": {
      "Adelaide": {
        "exemplarCity": "Adelaide"
      },
      "Brisbane": {
        "exemplarCity": "Brisbane"
      },
      "Broken_Hill": {
        "exemplarCity": "Broken Hill"
      },
      "Currie": {
        "exemplarCity": "Currie"
      },
      "Darwin": {
        "exemplarCity": "Darwin"
      },
      "Eucla": {
        "exemplarCity": "Eucla"
      }
    }
  }
}
```

```
    },  
    "Hobart": {  
      "exemplarCity": "Hobart"  
    },  
    "Lindeman": {  
      "exemplarCity": "Lindeman"  
    },  
    "Lord_Howe": {  
      "exemplarCity": "Lord Howe"  
    },  
    "Melbourne": {  
      "exemplarCity": "Melbourne"  
    },  
    "Perth": {  
      "exemplarCity": "Perth"  
    },  
    "Sydney": {  
      "exemplarCity": "Sydney"  
    }  
  },  
  "Pacific": {  
    "Apia": {  
      "exemplarCity": "Apia"  
    },  
    "Auckland": {  
      "exemplarCity": "Auckland"  
    },  
    "Bougainville": {  
      "exemplarCity": "Bougainville"  
    },  
    "Chatham": {  
      "exemplarCity": "Chatham"  
    },  
    "Easter": {  
      "exemplarCity": "Osterinsel"  
    },  
    "Efate": {  
      "exemplarCity": "Efate"  
    },  
    "Enderbury": {  
      "exemplarCity": "Enderbury"  
    },  
    "Fakaofu": {  
      "exemplarCity": "Fakaofu"  
    },  
    "Fiji": {  
      "exemplarCity": "Fidschi"  
    },  
    "Funafuti": {  
      "exemplarCity": "Funafuti"  
    },  
    "Galapagos": {  
      "exemplarCity": "Galapagos"  
    },  
    "Gambier": {  
      "exemplarCity": "Gambier"  
    },  
  },  
}
```

```
"Guadalcanal": {
  "exemplarCity": "Guadalcanal"
},
"Guam": {
  "exemplarCity": "Guam"
},
"Honolulu": {
  "exemplarCity": "Honolulu"
},
"Johnston": {
  "exemplarCity": "Johnston"
},
"Kiritimati": {
  "exemplarCity": "Kiritimati"
},
"Kosrae": {
  "exemplarCity": "Kosrae"
},
"Kwajalein": {
  "exemplarCity": "Kwajalein"
},
"Majuro": {
  "exemplarCity": "Majuro"
},
"Marquesas": {
  "exemplarCity": "Marquesas"
},
"Midway": {
  "exemplarCity": "Midway"
},
"Nauru": {
  "exemplarCity": "Nauru"
},
"Niue": {
  "exemplarCity": "Niue"
},
"Norfolk": {
  "exemplarCity": "Norfolk"
},
"Noumea": {
  "exemplarCity": "Noumea"
},
"Pago_Pago": {
  "exemplarCity": "Pago Pago"
},
"Palau": {
  "exemplarCity": "Palau"
},
"Pitcairn": {
  "exemplarCity": "Pitcairn"
},
"Ponape": {
  "exemplarCity": "Pohnpei"
},
"Port_Moresby": {
  "exemplarCity": "Port Moresby"
},
}
```

```
"Rarotonga": {
  "exemplarCity": "Rarotonga"
},
"Saipan": {
  "exemplarCity": "Saipan"
},
"Tahiti": {
  "exemplarCity": "Tahiti"
},
"Tarawa": {
  "exemplarCity": "Tarawa"
},
"Tongatapu": {
  "exemplarCity": "Tongatapu"
},
"Truk": {
  "exemplarCity": "Chuuk"
},
"Wake": {
  "exemplarCity": "Wake"
},
"Wallis": {
  "exemplarCity": "Wallis"
}
},
"Arctic": {
  "Longyearbyen": {
    "exemplarCity": "Longyearbyen"
  }
},
"Antarctica": {
  "Casey": {
    "exemplarCity": "Casey"
  },
  "Davis": {
    "exemplarCity": "Davis"
  },
  "DumontDUrville": {
    "exemplarCity": "Dumont d'Urville"
  },
  "Macquarie": {
    "exemplarCity": "Macquarie"
  },
  "Mawson": {
    "exemplarCity": "Mawson"
  },
  "McMurdo": {
    "exemplarCity": "McMurdo"
  },
  "Palmer": {
    "exemplarCity": "Palmer"
  },
  "Rothera": {
    "exemplarCity": "Rothera"
  },
  "Syowa": {
    "exemplarCity": "Syowa"
  }
}
```

```
    },  
    "Troll": {  
      "exemplarCity": "Troll"  
    },  
    "Vostok": {  
      "exemplarCity": "Wostok"  
    }  
  },  
  "Etc": {  
    "GMT": {  
      "exemplarCity": "GMT"  
    },  
    "GMT1": {  
      "exemplarCity": "GMT+1"  
    },  
    "GMT10": {  
      "exemplarCity": "GMT+10"  
    },  
    "GMT11": {  
      "exemplarCity": "GMT+11"  
    },  
    "GMT12": {  
      "exemplarCity": "GMT+12"  
    },  
    "GMT2": {  
      "exemplarCity": "GMT+2"  
    },  
    "GMT3": {  
      "exemplarCity": "GMT+3"  
    },  
    "GMT4": {  
      "exemplarCity": "GMT+4"  
    },  
    "GMT5": {  
      "exemplarCity": "GMT+5"  
    },  
    "GMT6": {  
      "exemplarCity": "GMT+6"  
    },  
    "GMT7": {  
      "exemplarCity": "GMT+7"  
    },  
    "GMT8": {  
      "exemplarCity": "GMT+8"  
    },  
    "GMT9": {  
      "exemplarCity": "GMT+9"  
    },  
    "GMT-1": {  
      "exemplarCity": "GMT-1"  
    },  
    "GMT-10": {  
      "exemplarCity": "GMT-10"  
    },  
    "GMT-11": {  
      "exemplarCity": "GMT-11"  
    }  
  },  
}
```



```

    "GMT-12": {
      "exemplarCity": "GMT-12"
    },
    "GMT-13": {
      "exemplarCity": "GMT-13"
    },
    "GMT-14": {
      "exemplarCity": "GMT-14"
    },
    "GMT-2": {
      "exemplarCity": "GMT-2"
    },
    "GMT-3": {
      "exemplarCity": "GMT-3"
    },
    "GMT-4": {
      "exemplarCity": "GMT-4"
    },
    "GMT-5": {
      "exemplarCity": "GMT-5"
    },
    "GMT-6": {
      "exemplarCity": "GMT-6"
    },
    "GMT-7": {
      "exemplarCity": "GMT-7"
    },
    "GMT-8": {
      "exemplarCity": "GMT-8"
    },
    "GMT-9": {
      "exemplarCity": "GMT-9"
    },
    "Unknown": {
      "exemplarCity": "Unbekannt"
    }
  },
  "metazone": {
    "Acre": {
      "long": {
        "generic": "Acre-Zeit",
        "standard": "Acre-Normalzeit",
        "daylight": "Acre-Sommerzeit"
      }
    },
    "Afghanistan": {
      "long": {
        "standard": "Afghanistan-Zeit"
      }
    },
    "Africa_Central": {
      "long": {
        "standard": "Zentralafrikanische Zeit"
      }
    },
    "Africa_Eastern": {

```

```

        "long": {
            "standard": "Ostafrikanische Zeit"
        }
    },
    "Africa_Southern": {
        "long": {
            "standard": "Südafrikanische Zeit"
        }
    },
    "Africa_Western": {
        "long": {
            "generic": "Westafrikanische Zeit",
            "standard": "Westafrikanische Normalzeit",
            "daylight": "Westafrikanische Sommerzeit"
        }
    },
    "Alaska": {
        "long": {
            "generic": "Alaska-Zeit",
            "standard": "Alaska-Normalzeit",
            "daylight": "Alaska-Sommerzeit"
        }
    },
    "Almaty": {
        "long": {
            "generic": "Almaty-Zeit",
            "standard": "Almaty-Normalzeit",
            "daylight": "Almaty-Sommerzeit"
        }
    },
    "Amazon": {
        "long": {
            "generic": "Amazonas-Zeit",
            "standard": "Amazonas-Normalzeit",
            "daylight": "Amazonas-Sommerzeit"
        }
    },
    "America_Central": {
        "long": {
            "generic": "Nordamerikanische Inlandzeit",
            "standard": "Nordamerikanische Inland-Normalzeit",
            "daylight": "Nordamerikanische Inland-Sommerzeit"
        }
    },
    "America_Eastern": {
        "long": {
            "generic": "Nordamerikanische Ostküstenzeit",
            "standard": "Nordamerikanische Ostküsten-Normalzeit",
            "daylight": "Nordamerikanische Ostküsten-Sommerzeit"
        }
    },
    "America_Mountain": {
        "long": {
            "generic": "Rocky-Mountain-Zeit",
            "standard": "Rocky Mountain-Normalzeit",
            "daylight": "Rocky-Mountain-Sommerzeit"
        }
    }
}

```

```

    },
    "America_Pacific": {
      "long": {
        "generic": "Nordamerikanische Westküstenzeit",
        "standard": "Nordamerikanische Westküsten-Normalzeit",
        "daylight": "Nordamerikanische Westküsten-Sommerzeit"
      }
    },
    "Anadyr": {
      "long": {
        "generic": "Anadyr Zeit",
        "standard": "Anadyr Normalzeit",
        "daylight": "Anadyr Sommerzeit"
      }
    },
    "Apia": {
      "long": {
        "generic": "Apia-Zeit",
        "standard": "Apia-Normalzeit",
        "daylight": "Apia-Sommerzeit"
      }
    },
    "Aqtai": {
      "long": {
        "generic": "Aqtai-Zeit",
        "standard": "Aqtai-Normalzeit",
        "daylight": "Aqtai-Sommerzeit"
      }
    },
    "Aqtobe": {
      "long": {
        "generic": "Aqtöbe-Zeit",
        "standard": "Aqtöbe-Normalzeit",
        "daylight": "Aqtöbe-Sommerzeit"
      }
    },
    "Arabian": {
      "long": {
        "generic": "Arabische Zeit",
        "standard": "Arabische Normalzeit",
        "daylight": "Arabische Sommerzeit"
      }
    },
    "Argentina": {
      "long": {
        "generic": "Argentinische Zeit",
        "standard": "Argentinische Normalzeit",
        "daylight": "Argentinische Sommerzeit"
      }
    },
    "Argentina_Western": {
      "long": {
        "generic": "Westargentinische Zeit",
        "standard": "Westargentinische Normalzeit",
        "daylight": "Westargentinische Sommerzeit"
      }
    }
  },

```

```
"Armenia": {
  "long": {
    "generic": "Armenische Zeit",
    "standard": "Armenische Normalzeit",
    "daylight": "Armenische Sommerzeit"
  }
},
"Atlantic": {
  "long": {
    "generic": "Atlantik-Zeit",
    "standard": "Atlantik-Normalzeit",
    "daylight": "Atlantik-Sommerzeit"
  }
},
"Australia_Central": {
  "long": {
    "generic": "Zentralaustralische Zeit",
    "standard": "Zentralaustralische Normalzeit",
    "daylight": "Zentralaustralische Sommerzeit"
  }
},
"Australia_CentralWestern": {
  "long": {
    "generic": "Zentral-/Westaustralische Zeit",
    "standard": "Zentral-/Westaustralische Normalzeit",
    "daylight": "Zentral-/Westaustralische Sommerzeit"
  }
},
"Australia_Eastern": {
  "long": {
    "generic": "Ostaustralische Zeit",
    "standard": "Ostaustralische Normalzeit",
    "daylight": "Ostaustralische Sommerzeit"
  }
},
"Australia_Western": {
  "long": {
    "generic": "Westaustralische Zeit",
    "standard": "Westaustralische Normalzeit",
    "daylight": "Westaustralische Sommerzeit"
  }
},
"Azerbaijan": {
  "long": {
    "generic": "Aserbaidshanische Zeit",
    "standard": "Aserbeidschanische Normalzeit",
    "daylight": "Aserbaidshanische Sommerzeit"
  }
},
"Azores": {
  "long": {
    "generic": "Azoren-Zeit",
    "standard": "Azoren-Normalzeit",
    "daylight": "Azoren-Sommerzeit"
  }
},
"Bangladesh": {
```

```
        "long": {
            "generic": "Bangladesch-Zeit",
            "standard": "Bangladesch-Normalzeit",
            "daylight": "Bangladesch-Sommerzeit"
        }
    },
    "Bhutan": {
        "long": {
            "standard": "Bhutan-Zeit"
        }
    },
    "Bolivia": {
        "long": {
            "standard": "Bolivianische Zeit"
        }
    },
    "Brasilia": {
        "long": {
            "generic": "Brasília-Zeit",
            "standard": "Brasília-Normalzeit",
            "daylight": "Brasília-Sommerzeit"
        }
    },
    "Brunei": {
        "long": {
            "standard": "Brunei-Zeit"
        }
    },
    "Cape_Verde": {
        "long": {
            "generic": "Cabo-Verde-Zeit",
            "standard": "Cabo-Verde-Normalzeit",
            "daylight": "Cabo-Verde-Sommerzeit"
        }
    },
    "Casey": {
        "long": {
            "standard": "Casey-Zeit"
        }
    },
    "Chamorro": {
        "long": {
            "standard": "Chamorro-Zeit"
        }
    },
    "Chatham": {
        "long": {
            "generic": "Chatham-Zeit",
            "standard": "Chatham-Normalzeit",
            "daylight": "Chatham-Sommerzeit"
        }
    },
    "Chile": {
        "long": {
            "generic": "Chilenische Zeit",
            "standard": "Chilenische Normalzeit",
            "daylight": "Chilenische Sommerzeit"
        }
    }
}
```

```
    }
  },
  "China": {
    "long": {
      "generic": "Chinesische Zeit",
      "standard": "Chinesische Normalzeit",
      "daylight": "Chinesische Sommerzeit"
    }
  },
  "Choibalsan": {
    "long": {
      "generic": "Tschoibalsan-Zeit",
      "standard": "Tschoibalsan-Normalzeit",
      "daylight": "Tschoibalsan-Sommerzeit"
    }
  },
  "Christmas": {
    "long": {
      "standard": "Weihnachtsinsel-Zeit"
    }
  },
  "Cocos": {
    "long": {
      "standard": "Kokosinseln-Zeit"
    }
  },
  "Colombia": {
    "long": {
      "generic": "Kolumbianische Zeit",
      "standard": "Kolumbianische Normalzeit",
      "daylight": "Kolumbianische Sommerzeit"
    }
  },
  "Cook": {
    "long": {
      "generic": "Cookinseln-Zeit",
      "standard": "Cookinseln-Normalzeit",
      "daylight": "Cookinseln-Sommerzeit"
    }
  },
  "Cuba": {
    "long": {
      "generic": "Kubanische Zeit",
      "standard": "Kubanische Normalzeit",
      "daylight": "Kubanische Sommerzeit"
    }
  },
  "Davis": {
    "long": {
      "standard": "Davis-Zeit"
    }
  },
  "DumontDUrville": {
    "long": {
      "standard": "Dumont-d'Urville-Zeit"
    }
  },
},
```

```
"East_Timor": {
  "long": {
    "standard": "Osttimor-Zeit"
  }
},
"Easter": {
  "long": {
    "generic": "Osterinsel-Zeit",
    "standard": "Osterinsel-Normalzeit",
    "daylight": "Osterinsel-Sommerzeit"
  }
},
"Ecuador": {
  "long": {
    "standard": "Ecuadorianische Zeit"
  }
},
"Europe_Central": {
  "long": {
    "generic": "Mitteleuropäische Zeit",
    "standard": "Mitteleuropäische Normalzeit",
    "daylight": "Mitteleuropäische Sommerzeit"
  },
  "short": {
    "generic": "MEZ",
    "standard": "MEZ",
    "daylight": "MESZ"
  }
},
"Europe_Eastern": {
  "long": {
    "generic": "Osteuropäische Zeit",
    "standard": "Osteuropäische Normalzeit",
    "daylight": "Osteuropäische Sommerzeit"
  },
  "short": {
    "generic": "OEZ",
    "standard": "OEZ",
    "daylight": "OESZ"
  }
},
"Europe_Further_Eastern": {
  "long": {
    "standard": "Kaliningrader Zeit"
  }
},
"Europe_Western": {
  "long": {
    "generic": "Westeuropäische Zeit",
    "standard": "Westeuropäische Normalzeit",
    "daylight": "Westeuropäische Sommerzeit"
  },
  "short": {
    "generic": "WEZ",
    "standard": "WEZ",
    "daylight": "WESZ"
  }
}
```

```
    },
    "Falkland": {
      "long": {
        "generic": "Falklandinseln-Zeit",
        "standard": "Falklandinseln-Normalzeit",
        "daylight": "Falklandinseln-Sommerzeit"
      }
    },
    "Fiji": {
      "long": {
        "generic": "Fidschi-Zeit",
        "standard": "Fidschi-Normalzeit",
        "daylight": "Fidschi-Sommerzeit"
      }
    },
    "French_Guiana": {
      "long": {
        "standard": "Französisch-Guayana-Zeit"
      }
    },
    "French_Southern": {
      "long": {
        "standard": "Französische Süd- und Antarktisgebiete-Zeit"
      }
    },
    "Galapagos": {
      "long": {
        "standard": "Galapagos-Zeit"
      }
    },
    "Gambier": {
      "long": {
        "standard": "Gambier-Zeit"
      }
    },
    "Georgia": {
      "long": {
        "generic": "Georgische Zeit",
        "standard": "Georgische Normalzeit",
        "daylight": "Georgische Sommerzeit"
      }
    },
    "Gilbert_Islands": {
      "long": {
        "standard": "Gilbert-Inseln-Zeit"
      }
    },
    "GMT": {
      "long": {
        "standard": "Mittlere Greenwich-Zeit"
      }
    },
    "Greenland_Eastern": {
      "long": {
        "generic": "Ostgrönland-Zeit",
        "standard": "Ostgrönland-Normalzeit",
        "daylight": "Ostgrönland-Sommerzeit"
      }
    }
  }
}
```



```
    },
    "Greenland_Western": {
      "long": {
        "generic": "Westgrönland-Zeit",
        "standard": "Westgrönland-Normalzeit",
        "daylight": "Westgrönland-Sommerzeit"
      }
    },
    "Guam": {
      "long": {
        "standard": "Guam-Zeit"
      }
    },
    "Gulf": {
      "long": {
        "standard": "Golf-Zeit"
      }
    },
    "Guyana": {
      "long": {
        "standard": "Guyana-Zeit"
      }
    },
    "Hawaii_Aleutian": {
      "long": {
        "generic": "Hawaii-Aleuten-Zeit",
        "standard": "Hawaii-Aleuten-Normalzeit",
        "daylight": "Hawaii-Aleuten-Sommerzeit"
      }
    },
    "Hong_Kong": {
      "long": {
        "generic": "Hongkong-Zeit",
        "standard": "Hongkong-Normalzeit",
        "daylight": "Hongkong-Sommerzeit"
      }
    },
    "Hovd": {
      "long": {
        "generic": "Chowd-Zeit",
        "standard": "Chowd-Normalzeit",
        "daylight": "Chowd-Sommerzeit"
      }
    },
    "India": {
      "long": {
        "standard": "Indische Zeit"
      }
    },
    "Indian_Ocean": {
      "long": {
        "standard": "Indischer Ozean-Zeit"
      }
    },
    "Indochina": {
      "long": {
```

```
        "standard": "Indochina-Zeit"
      },
    },
    "Indonesia_Central": {
      "long": {
        "standard": "Zentralindonesische Zeit"
      }
    },
    "Indonesia_Eastern": {
      "long": {
        "standard": "Ostindonesische Zeit"
      }
    },
    "Indonesia_Western": {
      "long": {
        "standard": "Westindonesische Zeit"
      }
    },
    "Iran": {
      "long": {
        "generic": "Iranische Zeit",
        "standard": "Iranische Normalzeit",
        "daylight": "Iranische Sommerzeit"
      }
    },
    "Irkutsk": {
      "long": {
        "generic": "Irkutsk-Zeit",
        "standard": "Irkutsk-Normalzeit",
        "daylight": "Irkutsk-Sommerzeit"
      }
    },
    "Israel": {
      "long": {
        "generic": "Israelische Zeit",
        "standard": "Israelische Normalzeit",
        "daylight": "Israelische Sommerzeit"
      }
    },
    "Japan": {
      "long": {
        "generic": "Japanische Zeit",
        "standard": "Japanische Normalzeit",
        "daylight": "Japanische Sommerzeit"
      }
    },
    "Kamchatka": {
      "long": {
        "generic": "Kamtschatka-Zeit",
        "standard": "Kamtschatka-Normalzeit",
        "daylight": "Kamtschatka-Sommerzeit"
      }
    },
    "Kazakhstan_Eastern": {
      "long": {
        "standard": "Ostkasachische Zeit"
      }
    }
  }
}
```

```
    },
    "Kazakhstan_Western": {
      "long": {
        "standard": "Westkasachische Zeit"
      }
    },
    "Korea": {
      "long": {
        "generic": "Koreanische Zeit",
        "standard": "Koreanische Normalzeit",
        "daylight": "Koreanische Sommerzeit"
      }
    },
    "Kosrae": {
      "long": {
        "standard": "Kosrae-Zeit"
      }
    },
    "Krasnoyarsk": {
      "long": {
        "generic": "Krasnojarsk-Zeit",
        "standard": "Krasnojarsk-Normalzeit",
        "daylight": "Krasnojarsk-Sommerzeit"
      }
    },
    "Kyrgystan": {
      "long": {
        "standard": "Kirgisistan-Zeit"
      }
    },
    "Lanka": {
      "long": {
        "standard": "Sri-Lanka-Zeit"
      }
    },
    "Line_Islands": {
      "long": {
        "standard": "Linieninseln-Zeit"
      }
    },
    "Lord_Howe": {
      "long": {
        "generic": "Lord-Howe-Zeit",
        "standard": "Lord-Howe-Normalzeit",
        "daylight": "Lord-Howe-Sommerzeit"
      }
    },
    "Macau": {
      "long": {
        "generic": "Macau-Zeit",
        "standard": "Macau-Normalzeit",
        "daylight": "Macau-Sommerzeit"
      }
    },
    "Macquarie": {
      "long": {
        "standard": "Macquarieinsel-Zeit"
```

```
    }
  },
  "Magadan": {
    "long": {
      "generic": "Magadan-Zeit",
      "standard": "Magadan-Normalzeit",
      "daylight": "Magadan-Sommerzeit"
    }
  },
  "Malaysia": {
    "long": {
      "standard": "Malaysische Zeit"
    }
  },
  "Maldives": {
    "long": {
      "standard": "Malediven-Zeit"
    }
  },
  "Marquesas": {
    "long": {
      "standard": "Marquesas-Zeit"
    }
  },
  "Marshall_Islands": {
    "long": {
      "standard": "Marshallinseln-Zeit"
    }
  },
  "Mauritius": {
    "long": {
      "generic": "Mauritius-Zeit",
      "standard": "Mauritius-Normalzeit",
      "daylight": "Mauritius-Sommerzeit"
    }
  },
  "Mawson": {
    "long": {
      "standard": "Mawson-Zeit"
    }
  },
  "Mexico_Northwest": {
    "long": {
      "generic": "Mexiko Nordwestliche Zone-Zeit",
      "standard": "Mexiko Nordwestliche Zone-Normalzeit",
      "daylight": "Mexiko Nordwestliche Zone-Sommerzeit"
    }
  },
  "Mexico_Pacific": {
    "long": {
      "generic": "Mexiko Pazifikzone-Zeit",
      "standard": "Mexiko Pazifikzone-Normalzeit",
      "daylight": "Mexiko Pazifikzone-Sommerzeit"
    }
  },
  "Mongolia": {
    "long": {
```

```
        "generic": "Ulaanbaatar-Zeit",
        "standard": "Ulaanbaatar-Normalzeit",
        "daylight": "Ulaanbaatar-Sommerzeit"
    }
},
"Moscow": {
    "long": {
        "generic": "Moskauer Zeit",
        "standard": "Moskauer Normalzeit",
        "daylight": "Moskauer Sommerzeit"
    }
},
"Myanmar": {
    "long": {
        "standard": "Myanmar-Zeit"
    }
},
"Nauru": {
    "long": {
        "standard": "Nauru-Zeit"
    }
},
"Nepal": {
    "long": {
        "standard": "Nepalesische Zeit"
    }
},
"New_Caledonia": {
    "long": {
        "generic": "Neukaledonische Zeit",
        "standard": "Neukaledonische Normalzeit",
        "daylight": "Neukaledonische Sommerzeit"
    }
},
"New_Zealand": {
    "long": {
        "generic": "Neuseeland-Zeit",
        "standard": "Neuseeland-Normalzeit",
        "daylight": "Neuseeland-Sommerzeit"
    }
},
"Newfoundland": {
    "long": {
        "generic": "Neufundland-Zeit",
        "standard": "Neufundland-Normalzeit",
        "daylight": "Neufundland-Sommerzeit"
    }
},
"Niue": {
    "long": {
        "standard": "Niue-Zeit"
    }
},
"Norfolk": {
    "long": {
        "standard": "Norfolkinsel-Zeit"
    }
}
```

```

    },
    "Noronha": {
      "long": {
        "generic": "Fernando de Noronha-Zeit",
        "standard": "Fernando de Noronha-Normalzeit",
        "daylight": "Fernando de Noronha-Sommerzeit"
      }
    },
    "North_Mariana": {
      "long": {
        "standard": "Nördliche-Marianen-Zeit"
      }
    },
    "Novosibirsk": {
      "long": {
        "generic": "Nowosibirsk-Zeit",
        "standard": "Nowosibirsk-Normalzeit",
        "daylight": "Nowosibirsk-Sommerzeit"
      }
    },
    "Omsk": {
      "long": {
        "generic": "Omsk-Zeit",
        "standard": "Omsk-Normalzeit",
        "daylight": "Omsk-Sommerzeit"
      }
    },
    "Pakistan": {
      "long": {
        "generic": "Pakistanische Zeit",
        "standard": "Pakistanische Normalzeit",
        "daylight": "Pakistanische Sommerzeit"
      }
    },
    "Palau": {
      "long": {
        "standard": "Palau-Zeit"
      }
    },
    "Papua_New_Guinea": {
      "long": {
        "standard": "Papua-Neuguinea-Zeit"
      }
    },
    "Paraguay": {
      "long": {
        "generic": "Paraguayische Zeit",
        "standard": "Paraguayische Normalzeit",
        "daylight": "Paraguayische Sommerzeit"
      }
    },
    "Peru": {
      "long": {
        "generic": "Peruanische Zeit",
        "standard": "Peruanische Normalzeit",
        "daylight": "Peruanische Sommerzeit"
      }
    }
  }

```

```
    },
    "Philippines": {
      "long": {
        "generic": "Philippinische Zeit",
        "standard": "Philippinische Normalzeit",
        "daylight": "Philippinische Sommerzeit"
      }
    },
    "Phoenix_Islands": {
      "long": {
        "standard": "Phoenixinseln-Zeit"
      }
    },
    "Pierre_Miquelon": {
      "long": {
        "generic": "Saint-Pierre-und-Miquelon-Zeit",
        "standard": "Saint-Pierre-und-Miquelon-Normalzeit",
        "daylight": "Saint-Pierre-und-Miquelon-Sommerzeit"
      }
    },
    "Pitcairn": {
      "long": {
        "standard": "Pitcairninseln-Zeit"
      }
    },
    "Ponape": {
      "long": {
        "standard": "Ponape-Zeit"
      }
    },
    "Pyongyang": {
      "long": {
        "standard": "Pjöngjang-Zeit"
      }
    },
    "Qyzylorda": {
      "long": {
        "generic": "Quysylorda-Zeit",
        "standard": "Quysylorda-Normalzeit",
        "daylight": "Qysylorda-Sommerzeit"
      }
    },
    "Reunion": {
      "long": {
        "standard": "Réunion-Zeit"
      }
    },
    "Rothera": {
      "long": {
        "standard": "Rothera-Zeit"
      }
    },
    "Sakhalin": {
      "long": {
        "generic": "Sachalin-Zeit",
        "standard": "Sachalin-Normalzeit",
        "daylight": "Sachalin-Sommerzeit"
      }
    }
  }
```

```

    }
  },
  "Samara": {
    "long": {
      "generic": "Samara-Zeit",
      "standard": "Samara-Normalzeit",
      "daylight": "Samara-Sommerzeit"
    }
  },
  "Samoa": {
    "long": {
      "generic": "Samoa-Zeit",
      "standard": "Samoa-Normalzeit",
      "daylight": "Samoa-Sommerzeit"
    }
  },
  "Seychelles": {
    "long": {
      "standard": "Seychellen-Zeit"
    }
  },
  "Singapore": {
    "long": {
      "standard": "Singapur-Zeit"
    }
  },
  "Solomon": {
    "long": {
      "standard": "Salomoninseln-Zeit"
    }
  },
  "South_Georgia": {
    "long": {
      "standard": "Südgeorgische Zeit"
    }
  },
  "Suriname": {
    "long": {
      "standard": "Suriname-Zeit"
    }
  },
  "Syowa": {
    "long": {
      "standard": "Syowa-Zeit"
    }
  },
  "Tahiti": {
    "long": {
      "standard": "Tahiti-Zeit"
    }
  },
  "Taipei": {
    "long": {
      "generic": "Taipeh-Zeit",
      "standard": "Taipeh-Normalzeit",
      "daylight": "Taipeh-Sommerzeit"
    }
  }
}

```



```
    },
    "Tajikistan": {
      "long": {
        "standard": "Tadschikistan-Zeit"
      }
    },
    "Tokelau": {
      "long": {
        "standard": "Tokelau-Zeit"
      }
    },
    "Tonga": {
      "long": {
        "generic": "Tonganische Zeit",
        "standard": "Tonganische Normalzeit",
        "daylight": "Tonganische Sommerzeit"
      }
    },
    "Truk": {
      "long": {
        "standard": "Chuuk-Zeit"
      }
    },
    "Turkmenistan": {
      "long": {
        "generic": "Turkmenistan-Zeit",
        "standard": "Turkmenistan-Normalzeit",
        "daylight": "Turkmenistan-Sommerzeit"
      }
    },
    "Tuvalu": {
      "long": {
        "standard": "Tuvalu-Zeit"
      }
    },
    "Uruguay": {
      "long": {
        "generic": "Uruguayanische Zeit",
        "standard": "Uruguyanische Normalzeit",
        "daylight": "Uruguayanische Sommerzeit"
      }
    },
    "Uzbekistan": {
      "long": {
        "generic": "Usbekistan-Zeit",
        "standard": "Usbekistan-Normalzeit",
        "daylight": "Usbekistan-Sommerzeit"
      }
    },
    "Vanuatu": {
      "long": {
        "generic": "Vanuatu-Zeit",
        "standard": "Vanuatu-Normalzeit",
        "daylight": "Vanuatu-Sommerzeit"
      }
    },
    "Venezuela": {
```

```

        "long": {
            "standard": "Venezuela-Zeit"
        },
        "Vladivostok": {
            "long": {
                "generic": "Wladiwostok-Zeit",
                "standard": "Wladiwostok-Normalzeit",
                "daylight": "Wladiwostok-Sommerzeit"
            }
        },
        "Volgograd": {
            "long": {
                "generic": "Wolgograd-Zeit",
                "standard": "Wolgograd-Normalzeit",
                "daylight": "Wolgograd-Sommerzeit"
            }
        },
        "Vostok": {
            "long": {
                "standard": "Wostok-Zeit"
            }
        },
        "Wake": {
            "long": {
                "standard": "Wake-Insel-Zeit"
            }
        },
        "Wallis": {
            "long": {
                "standard": "Wallis-und-Futuna-Zeit"
            }
        },
        "Yakutsk": {
            "long": {
                "generic": "Jakutsk-Zeit",
                "standard": "Jakutsk-Normalzeit",
                "daylight": "Jakutsk-Sommerzeit"
            }
        },
        "Yekaterinburg": {
            "long": {
                "generic": "Jekaterinburg-Zeit",
                "standard": "Jekaterinburg-Normalzeit",
                "daylight": "Jekaterinburg-Sommerzeit"
            }
        }
    }
}

```

TIMEZONENAMES.JSX

```

{
  "main";
  {
    "de";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 12879 $",
          "_cldrVersion";
          "30.0.3";
        }
        "language";
        "de";
      }
    }
    "dates";
    {
      "timeZoneNames";
      {
        "hourFormat";
        "+HH:mm;-HH:mm",
        "gmtFormat";
        "GMT{0}",
        "gmtZeroFormat";
        "GMT",
        "regionFormat";
        "{0} Zeit",
        "regionFormat-type-daylight";
        "{0} Sommerzeit",
        "regionFormat-type-standard";
        "{0} Normalzeit",
        "fallbackFormat";
        "{1} ({0})",
        "zone";
        {
          "America";
          {
            "Adak";
            {
              "exemplarCity";
              "Adak";
            }
            "Anchorage";
            {
              "exemplarCity";
              "Anchorage";
            }
            "Anguilla";
            {
              "exemplarCity";
              "Anguilla";
            }
            "Antigua";
            {
              "exemplarCity";
            }
          }
        }
      }
    }
  }
}

```

```
        "Antigua";
    }
    "Araguaina";
    {
        "exemplarCity";
        "Araguaina";
    }
    "Argentina";
    {
        "Rio_Gallegos";
        {
            "exemplarCity";
            "Rio Gallegos";
        }
        "San_Juan";
        {
            "exemplarCity";
            "San Juan";
        }
        "Ushuaia";
        {
            "exemplarCity";
            "Ushuaia";
        }
        "La_Rioja";
        {
            "exemplarCity";
            "La Rioja";
        }
        "San_Luis";
        {
            "exemplarCity";
            "San Luis";
        }
        "Salta";
        {
            "exemplarCity";
            "Salta";
        }
        "Tucuman";
        {
            "exemplarCity";
            "Tucuman";
        }
    }
    "Aruba";
    {
        "exemplarCity";
        "Aruba";
    }
    "Asuncion";
    {
        "exemplarCity";
        "Asunción";
    }
    "Bahia";
    {
```

```
        "exemplarCity";
        "Bahia";
    }
    "Bahia_Banderas";
    {
        "exemplarCity";
        "Bahia Banderas";
    }
    "Barbados";
    {
        "exemplarCity";
        "Barbados";
    }
    "Belem";
    {
        "exemplarCity";
        "Belem";
    }
    "Belize";
    {
        "exemplarCity";
        "Belize";
    }
    "Blanc-Sablon";
    {
        "exemplarCity";
        "Blanc-Sablon";
    }
    "Boa_Vista";
    {
        "exemplarCity";
        "Boa Vista";
    }
    "Bogota";
    {
        "exemplarCity";
        "Bogotá";
    }
    "Boise";
    {
        "exemplarCity";
        "Boise";
    }
    "Buenos_Aires";
    {
        "exemplarCity";
        "Buenos Aires";
    }
    "Cambridge_Bay";
    {
        "exemplarCity";
        "Cambridge Bay";
    }
    "Campo_Grande";
    {
        "exemplarCity";
        "Campo Grande";
    }
}
```

```
}
"Cancun";
{
  "exemplarCity";
  "Cancún";
}
"Caracas";
{
  "exemplarCity";
  "Caracas";
}
"Catamarca";
{
  "exemplarCity";
  "Catamarca";
}
"Cayenne";
{
  "exemplarCity";
  "Cayenne";
}
"Cayman";
{
  "exemplarCity";
  "Kaimaninseln";
}
"Chicago";
{
  "exemplarCity";
  "Chicago";
}
"Chihuahua";
{
  "exemplarCity";
  "Chihuahua";
}
"Coral_Harbour";
{
  "exemplarCity";
  "Atikokan";
}
"Cordoba";
{
  "exemplarCity";
  "Córdoba";
}
"Costa_Rica";
{
  "exemplarCity";
  "Costa Rica";
}
"Creston";
{
  "exemplarCity";
  "Creston";
}
"Cuiaba";
```

```
{
    "exemplarCity";
    "Cuiaba";
}
"Curacao";
{
    "exemplarCity";
    "Curaçao";
}
"Danmarkshavn";
{
    "exemplarCity";
    "Danmarkshavn";
}
"Dawson";
{
    "exemplarCity";
    "Dawson";
}
"Dawson_Creek";
{
    "exemplarCity";
    "Dawson Creek";
}
"Denver";
{
    "exemplarCity";
    "Denver";
}
"Detroit";
{
    "exemplarCity";
    "Detroit";
}
"Dominica";
{
    "exemplarCity";
    "Dominica";
}
"Edmonton";
{
    "exemplarCity";
    "Edmonton";
}
"Eirunepe";
{
    "exemplarCity";
    "Eirunepe";
}
"El_Salvador";
{
    "exemplarCity";
    "El Salvador";
}
"Fort_Nelson";
{
    "exemplarCity";
```

```
        "Fort Nelson";
    }
    "Fortaleza";
    {
        "exemplarCity";
        "Fortaleza";
    }
    "Glace_Bay";
    {
        "exemplarCity";
        "Glace Bay";
    }
    "Godthab";
    {
        "exemplarCity";
        "Nuuk";
    }
    "Goose_Bay";
    {
        "exemplarCity";
        "Goose Bay";
    }
    "Grand_Turk";
    {
        "exemplarCity";
        "Grand Turk";
    }
    "Grenada";
    {
        "exemplarCity";
        "Grenada";
    }
    "Guadeloupe";
    {
        "exemplarCity";
        "Guadeloupe";
    }
    "Guatemala";
    {
        "exemplarCity";
        "Guatemala";
    }
    "Guayaquil";
    {
        "exemplarCity";
        "Guayaquil";
    }
    "Guyana";
    {
        "exemplarCity";
        "Guyana";
    }
    "Halifax";
    {
        "exemplarCity";
        "Halifax";
    }
}
```



```
"Havana";
{
  "exemplarCity";
  "Havanna";
}
"Hermosillo";
{
  "exemplarCity";
  "Hermosillo";
}
"Indiana";
{
  "Vincennes";
  {
    "exemplarCity";
    "Vincennes, Indiana";
  }
  "Petersburg";
  {
    "exemplarCity";
    "Petersburg, Indiana";
  }
  "Tell_City";
  {
    "exemplarCity";
    "Tell City, Indiana";
  }
  "Knox";
  {
    "exemplarCity";
    "Knox, Indiana";
  }
  "Winamac";
  {
    "exemplarCity";
    "Winamac, Indiana";
  }
  "Marengo";
  {
    "exemplarCity";
    "Marengo, Indiana";
  }
  "Vevay";
  {
    "exemplarCity";
    "Vevay, Indiana";
  }
}
"Indianapolis";
{
  "exemplarCity";
  "Indianapolis";
}
"Inuvik";
{
  "exemplarCity";
  "Inuvik";
}
```

```
}
"Iqaluit";
{
  "exemplarCity";
  "Iqaluit";
}
"Jamaica";
{
  "exemplarCity";
  "Jamaika";
}
"Jujuy";
{
  "exemplarCity";
  "Jujuy";
}
"Juneau";
{
  "exemplarCity";
  "Juneau";
}
"Kentucky";
{
  "Monticello";
  {
    "exemplarCity";
    "Monticello, Kentucky";
  }
}
"Kralendijk";
{
  "exemplarCity";
  "Kralendijk";
}
"La_Paz";
{
  "exemplarCity";
  "La Paz";
}
"Lima";
{
  "exemplarCity";
  "Lima";
}
"Los_Angeles";
{
  "exemplarCity";
  "Los Angeles";
}
"Louisville";
{
  "exemplarCity";
  "Louisville";
}
"Lower_Princes";
{
  "exemplarCity";
```

```
        "Lower Prince's Quarter";
    }
    "Maceio";
    {
        "exemplarCity";
        "Maceio";
    }
    "Managua";
    {
        "exemplarCity";
        "Managua";
    }
    "Manaus";
    {
        "exemplarCity";
        "Manaus";
    }
    "Marigot";
    {
        "exemplarCity";
        "Marigot";
    }
    "Martinique";
    {
        "exemplarCity";
        "Martinique";
    }
    "Matamoros";
    {
        "exemplarCity";
        "Matamoros";
    }
    "Mazatlan";
    {
        "exemplarCity";
        "Mazatlan";
    }
    "Mendoza";
    {
        "exemplarCity";
        "Mendoza";
    }
    "Menominee";
    {
        "exemplarCity";
        "Menominee";
    }
    "Merida";
    {
        "exemplarCity";
        "Merida";
    }
    "Metlakatla";
    {
        "exemplarCity";
        "Metlakatla";
    }
}
```

```
"Mexico_City";
{
  "exemplarCity";
  "Mexiko-Stadt";
}
"Miquelon";
{
  "exemplarCity";
  "Miquelon";
}
"Moncton";
{
  "exemplarCity";
  "Moncton";
}
"Monterrey";
{
  "exemplarCity";
  "Monterrey";
}
"Montevideo";
{
  "exemplarCity";
  "Montevideo";
}
"Montserrat";
{
  "exemplarCity";
  "Montserrat";
}
"Nassau";
{
  "exemplarCity";
  "Nassau";
}
"New_York";
{
  "exemplarCity";
  "New York";
}
"Nipigon";
{
  "exemplarCity";
  "Nipigon";
}
"Nome";
{
  "exemplarCity";
  "Nome";
}
"Noronha";
{
  "exemplarCity";
  "Noronha";
}
"North_Dakota";
{
```

```
"Beulah";
{
  "exemplarCity";
  "Beulah, North Dakota";
}
"New_Salem";
{
  "exemplarCity";
  "New Salem, North Dakota";
}
"Center";
{
  "exemplarCity";
  "Center, North Dakota";
}
}
"Ojinaga";
{
  "exemplarCity";
  "Ojinaga";
}
"Panama";
{
  "exemplarCity";
  "Panama";
}
"Pangnirtung";
{
  "exemplarCity";
  "Pangnirtung";
}
"Paramaribo";
{
  "exemplarCity";
  "Paramaribo";
}
"Phoenix";
{
  "exemplarCity";
  "Phoenix";
}
"Port-au-Prince";
{
  "exemplarCity";
  "Port-au-Prince";
}
"Port_of_Spain";
{
  "exemplarCity";
  "Port of Spain";
}
"Porto_Velho";
{
  "exemplarCity";
  "Porto Velho";
}
}
"Puerto_Rico";
```

```
{
    "exemplarCity";
    "Puerto Rico";
}
"Rainy_River";
{
    "exemplarCity";
    "Rainy River";
}
"Rankin_Inlet";
{
    "exemplarCity";
    "Rankin Inlet";
}
"Recife";
{
    "exemplarCity";
    "Recife";
}
"Regina";
{
    "exemplarCity";
    "Regina";
}
"Resolute";
{
    "exemplarCity";
    "Resolute";
}
"Rio_Branco";
{
    "exemplarCity";
    "Rio Branco";
}
"Santa_Isabel";
{
    "exemplarCity";
    "Santa Isabel";
}
"Santarem";
{
    "exemplarCity";
    "Santarem";
}
"Santiago";
{
    "exemplarCity";
    "Santiago";
}
"Santo_Domingo";
{
    "exemplarCity";
    "Santo Domingo";
}
"Sao_Paulo";
{
    "exemplarCity";
```

```
        "São Paulo";
    }
    "Scoresbysund";
    {
        "exemplarCity";
        "Ittoqqortoormiit";
    }
    "Sitka";
    {
        "exemplarCity";
        "Sitka";
    }
    "St_Barthelemy";
    {
        "exemplarCity";
        "Saint-Barthélemy";
    }
    "St_Johns";
    {
        "exemplarCity";
        "St. John's";
    }
    "St_Kitts";
    {
        "exemplarCity";
        "St. Kitts";
    }
    "St_Lucia";
    {
        "exemplarCity";
        "St. Lucia";
    }
    "St_Thomas";
    {
        "exemplarCity";
        "St. Thomas";
    }
    "St_Vincent";
    {
        "exemplarCity";
        "St. Vincent";
    }
    "Swift_Current";
    {
        "exemplarCity";
        "Swift Current";
    }
    "Tegucigalpa";
    {
        "exemplarCity";
        "Tegucigalpa";
    }
    "Thule";
    {
        "exemplarCity";
        "Thule";
    }
}
```

```
        "Thunder_Bay";
        {
            "exemplarCity";
            "Thunder Bay";
        }
        "Tijuana";
        {
            "exemplarCity";
            "Tijuana";
        }
        "Toronto";
        {
            "exemplarCity";
            "Toronto";
        }
        "Tortola";
        {
            "exemplarCity";
            "Tortola";
        }
        "Vancouver";
        {
            "exemplarCity";
            "Vancouver";
        }
        "Whitehorse";
        {
            "exemplarCity";
            "Whitehorse";
        }
        "Winnipeg";
        {
            "exemplarCity";
            "Winnipeg";
        }
        "Yakutat";
        {
            "exemplarCity";
            "Yakutat";
        }
        "Yellowknife";
        {
            "exemplarCity";
            "Yellowknife";
        }
    }
    "Atlantic";
    {
        "Azores";
        {
            "exemplarCity";
            "Azoren";
        }
        "Bermuda";
        {
            "exemplarCity";
            "Bermudas";
        }
    }
}
```



```
}
"Canary";
{
  "exemplarCity";
  "Kanaren";
}
"Cape_Verde";
{
  "exemplarCity";
  "Cabo Verde";
}
"Faeroe";
{
  "exemplarCity";
  "Färöer";
}
"Madeira";
{
  "exemplarCity";
  "Madeira";
}
"Reykjavik";
{
  "exemplarCity";
  "Reykjavík";
}
"South_Georgia";
{
  "exemplarCity";
  "Südgeorgien";
}
"St_Helena";
{
  "exemplarCity";
  "St. Helena";
}
"Stanley";
{
  "exemplarCity";
  "Stanley";
}
}
"Europe";
{
  "Amsterdam";
  {
    "exemplarCity";
    "Amsterdam";
  }
  "Andorra";
  {
    "exemplarCity";
    "Andorra";
  }
  "Astrakhan";
  {
    "exemplarCity";
```

```
        "Astrachan";
    }
    "Athens";
    {
        "exemplarCity";
        "Athen";
    }
    "Belgrade";
    {
        "exemplarCity";
        "Belgrad";
    }
    "Berlin";
    {
        "exemplarCity";
        "Berlin";
    }
    "Bratislava";
    {
        "exemplarCity";
        "Bratislava";
    }
    "Brussels";
    {
        "exemplarCity";
        "Brüssel";
    }
    "Bucharest";
    {
        "exemplarCity";
        "Bukarest";
    }
    "Budapest";
    {
        "exemplarCity";
        "Budapest";
    }
    "Busingen";
    {
        "exemplarCity";
        "Büsingen";
    }
    "Chisinau";
    {
        "exemplarCity";
        "Kischinau";
    }
    "Copenhagen";
    {
        "exemplarCity";
        "Kopenhagen";
    }
    "Dublin";
    {
        "long";
        {
            "daylight";
```

```
        "Irische Sommerzeit";
    }
    "exemplarCity";
    "Dublin";
}
"Gibraltar";
{
    "exemplarCity";
    "Gibraltar";
}
"Guernsey";
{
    "exemplarCity";
    "Guernsey";
}
"Helsinki";
{
    "exemplarCity";
    "Helsinki";
}
"Isle_of_Man";
{
    "exemplarCity";
    "Isle of Man";
}
"Istanbul";
{
    "exemplarCity";
    "Istanbul";
}
"Jersey";
{
    "exemplarCity";
    "Jersey";
}
"Kaliningrad";
{
    "exemplarCity";
    "Kaliningrad";
}
"Kiev";
{
    "exemplarCity";
    "Kiew";
}
"Kirov";
{
    "exemplarCity";
    "Kirow";
}
"Lisbon";
{
    "exemplarCity";
    "Lissabon";
}
"Ljubljana";
{
```

```
        "exemplarCity";
        "Ljubljana";
    }
    "London";
    {
        "long";
        {
            "daylight";
            "Britische Sommerzeit";
        }
        "exemplarCity";
        "London";
    }
    "Luxembourg";
    {
        "exemplarCity";
        "Luxemburg";
    }
    "Madrid";
    {
        "exemplarCity";
        "Madrid";
    }
    "Malta";
    {
        "exemplarCity";
        "Malta";
    }
    "Mariehamn";
    {
        "exemplarCity";
        "Mariehamn";
    }
    "Minsk";
    {
        "exemplarCity";
        "Minsk";
    }
    "Monaco";
    {
        "exemplarCity";
        "Monaco";
    }
    "Moscow";
    {
        "exemplarCity";
        "Moskau";
    }
    "Oslo";
    {
        "exemplarCity";
        "Oslo";
    }
    "Paris";
    {
        "exemplarCity";
        "Paris";
    }
}
```

```
}
"Podgorica";
{
  "exemplarCity";
  "Podgorica";
}
"Prague";
{
  "exemplarCity";
  "Prag";
}
"Riga";
{
  "exemplarCity";
  "Riga";
}
"Rome";
{
  "exemplarCity";
  "Rom";
}
"Samara";
{
  "exemplarCity";
  "Samara";
}
"San_Marino";
{
  "exemplarCity";
  "San Marino";
}
"Sarajevo";
{
  "exemplarCity";
  "Sarajevo";
}
"Simferopol";
{
  "exemplarCity";
  "Simferopol";
}
"Skopje";
{
  "exemplarCity";
  "Skopje";
}
"Sofia";
{
  "exemplarCity";
  "Sofia";
}
"Stockholm";
{
  "exemplarCity";
  "Stockholm";
}
"Tallinn";
```

```
{
    "exemplarCity";
    "Tallinn";
}
"Tirane";
{
    "exemplarCity";
    "Tirana";
}
"Ulyanovsk";
{
    "exemplarCity";
    "Uljanowsk";
}
"Uzhgorod";
{
    "exemplarCity";
    "Uschgorod";
}
"Vaduz";
{
    "exemplarCity";
    "Vaduz";
}
"Vatican";
{
    "exemplarCity";
    "Vatikan";
}
"Vienna";
{
    "exemplarCity";
    "Wien";
}
"Vilnius";
{
    "exemplarCity";
    "Vilnius";
}
"Volgograd";
{
    "exemplarCity";
    "Wolgograd";
}
"Warsaw";
{
    "exemplarCity";
    "Warschau";
}
"Zagreb";
{
    "exemplarCity";
    "Zagreb";
}
"Zaporozhye";
{
    "exemplarCity";
```

```
        "Saporischja";
    }
    "Zurich";
    {
        "exemplarCity";
        "Zürich";
    }
}
"Africa";
{
    "Abidjan";
    {
        "exemplarCity";
        "Abidjan";
    }
    "Accra";
    {
        "exemplarCity";
        "Accra";
    }
    "Addis_Ababa";
    {
        "exemplarCity";
        "Addis Abeba";
    }
    "Algiers";
    {
        "exemplarCity";
        "Algier";
    }
    "Asmera";
    {
        "exemplarCity";
        "Asmara";
    }
    "Bamako";
    {
        "exemplarCity";
        "Bamako";
    }
    "Bangui";
    {
        "exemplarCity";
        "Bangui";
    }
    "Banjul";
    {
        "exemplarCity";
        "Banjul";
    }
    "Bissau";
    {
        "exemplarCity";
        "Bissau";
    }
    "Blantyre";
    {
```

```
        "exemplarCity";
        "Blantyre";
    }
    "Brazzaville";
    {
        "exemplarCity";
        "Brazzaville";
    }
    "Bujumbura";
    {
        "exemplarCity";
        "Bujumbura";
    }
    "Cairo";
    {
        "exemplarCity";
        "Kairo";
    }
    "Casablanca";
    {
        "exemplarCity";
        "Casablanca";
    }
    "Ceuta";
    {
        "exemplarCity";
        "Ceuta";
    }
    "Conakry";
    {
        "exemplarCity";
        "Conakry";
    }
    "Dakar";
    {
        "exemplarCity";
        "Dakar";
    }
    "Dar_es_Salaam";
    {
        "exemplarCity";
        "Daressalam";
    }
    "Djibouti";
    {
        "exemplarCity";
        "Dschibuti";
    }
    "Douala";
    {
        "exemplarCity";
        "Douala";
    }
    "El_Aaiun";
    {
        "exemplarCity";
        "El Aaiún";
    }
```



```
}
"Freetown";
{
  "exemplarCity";
  "Freetown";
}
"Gaborone";
{
  "exemplarCity";
  "Gaborone";
}
"Harare";
{
  "exemplarCity";
  "Harare";
}
"Johannesburg";
{
  "exemplarCity";
  "Johannesburg";
}
"Juba";
{
  "exemplarCity";
  "Juba";
}
"Kampala";
{
  "exemplarCity";
  "Kampala";
}
"Khartoum";
{
  "exemplarCity";
  "Khartum";
}
"Kigali";
{
  "exemplarCity";
  "Kigali";
}
"Kinshasa";
{
  "exemplarCity";
  "Kinshasa";
}
"Lagos";
{
  "exemplarCity";
  "Lagos";
}
"Libreville";
{
  "exemplarCity";
  "Libreville";
}
"Lome";
```

```
{
    "exemplarCity";
    "Lomé";
}
"Luanda";
{
    "exemplarCity";
    "Luanda";
}
"Lubumbashi";
{
    "exemplarCity";
    "Lubumbashi";
}
"Lusaka";
{
    "exemplarCity";
    "Lusaka";
}
"Malabo";
{
    "exemplarCity";
    "Malabo";
}
"Maputo";
{
    "exemplarCity";
    "Maputo";
}
"Maseru";
{
    "exemplarCity";
    "Maseru";
}
"Mbabane";
{
    "exemplarCity";
    "Mbabane";
}
"Mogadishu";
{
    "exemplarCity";
    "Mogadishu";
}
"Monrovia";
{
    "exemplarCity";
    "Monrovia";
}
"Nairobi";
{
    "exemplarCity";
    "Nairobi";
}
"Ndjamena";
{
    "exemplarCity";
```

```

        "N'Djamena";
    }
    "Niamey";
    {
        "exemplarCity";
        "Niamey";
    }
    "Nouakchott";
    {
        "exemplarCity";
        "Nouakchott";
    }
    "Ouagadougou";
    {
        "exemplarCity";
        "Ouagadougou";
    }
    "Porto-Novo";
    {
        "exemplarCity";
        "Porto Novo";
    }
    "Sao_Tome";
    {
        "exemplarCity";
        "São Tomé";
    }
    "Tripoli";
    {
        "exemplarCity";
        "Tripolis";
    }
    "Tunis";
    {
        "exemplarCity";
        "Tunis";
    }
    "Windhoek";
    {
        "exemplarCity";
        "Windhoek";
    }
}
"Asia";
{
    "Aden";
    {
        "exemplarCity";
        "Aden";
    }
    "Almaty";
    {
        "exemplarCity";
        "Almaty";
    }
    "Amman";
    {

```

```
        "exemplarCity";
        "Amman";
    }
    "Anadyr";
    {
        "exemplarCity";
        "Anadyr";
    }
    "Aqtau";
    {
        "exemplarCity";
        "Aqtau";
    }
    "Aqtobe";
    {
        "exemplarCity";
        "Aktobe";
    }
    "Ashgabat";
    {
        "exemplarCity";
        "Aşgabat";
    }
    "Baghdad";
    {
        "exemplarCity";
        "Bagdad";
    }
    "Bahrain";
    {
        "exemplarCity";
        "Bahrain";
    }
    "Baku";
    {
        "exemplarCity";
        "Baku";
    }
    "Bangkok";
    {
        "exemplarCity";
        "Bangkok";
    }
    "Barnaul";
    {
        "exemplarCity";
        "Barnaul";
    }
    "Beirut";
    {
        "exemplarCity";
        "Beirut";
    }
    "Bishkek";
    {
        "exemplarCity";
        "Bischkek";
    }
```

```
}
"Brunei";
{
  "exemplarCity";
  "Brunei";
}
"Calcutta";
{
  "exemplarCity";
  "Kalkutta";
}
"Chita";
{
  "exemplarCity";
  "Tschita";
}
"Choibalsan";
{
  "exemplarCity";
  "Tschoibalsan";
}
"Colombo";
{
  "exemplarCity";
  "Colombo";
}
"Damascus";
{
  "exemplarCity";
  "Damaskus";
}
"Dhaka";
{
  "exemplarCity";
  "Dhaka";
}
"Dili";
{
  "exemplarCity";
  "Dili";
}
"Dubai";
{
  "exemplarCity";
  "Dubai";
}
"Dushanbe";
{
  "exemplarCity";
  "Duschanbe";
}
"Gaza";
{
  "exemplarCity";
  "Gaza";
}
"Hebron";
```

```
{
    "exemplarCity";
    "Hebron";
}
"Hong_Kong";
{
    "exemplarCity";
    "Hongkong";
}
"Hovd";
{
    "exemplarCity";
    "Chowd";
}
"Irkutsk";
{
    "exemplarCity";
    "Irkutsk";
}
"Jakarta";
{
    "exemplarCity";
    "Jakarta";
}
"Jayapura";
{
    "exemplarCity";
    "Jayapura";
}
"Jerusalem";
{
    "exemplarCity";
    "Jerusalem";
}
"Kabul";
{
    "exemplarCity";
    "Kabul";
}
"Kamchatka";
{
    "exemplarCity";
    "Kamtschatka";
}
"Karachi";
{
    "exemplarCity";
    "Karatschi";
}
"Katmandu";
{
    "exemplarCity";
    "Kathmandu";
}
"Khandyga";
{
    "exemplarCity";
```

```
        "Chandyga";
    }
    "Krasnoyarsk";
    {
        "exemplarCity";
        "Krasnojarsk";
    }
    "Kuala_Lumpur";
    {
        "exemplarCity";
        "Kuala Lumpur";
    }
    "Kuching";
    {
        "exemplarCity";
        "Kuching";
    }
    "Kuwait";
    {
        "exemplarCity";
        "Kuwait";
    }
    "Macau";
    {
        "exemplarCity";
        "Macao";
    }
    "Magadan";
    {
        "exemplarCity";
        "Magadan";
    }
    "Makassar";
    {
        "exemplarCity";
        "Makassar";
    }
    "Manila";
    {
        "exemplarCity";
        "Manila";
    }
    "Muscat";
    {
        "exemplarCity";
        "Maskat";
    }
    "Nicosia";
    {
        "exemplarCity";
        "Nikosia";
    }
    "Novokuznetsk";
    {
        "exemplarCity";
        "Nowokuznetsk";
    }
}
```

```
"Novosibirsk";
{
  "exemplarCity";
  "Nowosibirsk";
}
"Omsk";
{
  "exemplarCity";
  "Omsk";
}
"Oral";
{
  "exemplarCity";
  "Oral";
}
"Phnom_Penh";
{
  "exemplarCity";
  "Phnom Penh";
}
"Pontianak";
{
  "exemplarCity";
  "Pontianak";
}
"Pyongyang";
{
  "exemplarCity";
  "Pjöngjang";
}
"Qatar";
{
  "exemplarCity";
  "Katar";
}
"Qyzylorda";
{
  "exemplarCity";
  "Qysylorda";
}
"Rangoon";
{
  "exemplarCity";
  "Rangun";
}
"Riyadh";
{
  "exemplarCity";
  "Riad";
}
"Saigon";
{
  "exemplarCity";
  "Ho-Chi-Minh-Stadt";
}
"Sakhalin";
{
```



```
        "exemplarCity";
        "Sachalin";
    }
    "Samarkand";
    {
        "exemplarCity";
        "Samarkand";
    }
    "Seoul";
    {
        "exemplarCity";
        "Seoul";
    }
    "Shanghai";
    {
        "exemplarCity";
        "Shanghai";
    }
    "Singapore";
    {
        "exemplarCity";
        "Singapur";
    }
    "Srednekolymsk";
    {
        "exemplarCity";
        "Srednekolymsk";
    }
    "Taipei";
    {
        "exemplarCity";
        "Taipeh";
    }
    "Tashkent";
    {
        "exemplarCity";
        "Taschkent";
    }
    "Tbilisi";
    {
        "exemplarCity";
        "Tiflis";
    }
    "Tehran";
    {
        "exemplarCity";
        "Teheran";
    }
    "Thimphu";
    {
        "exemplarCity";
        "Thimphu";
    }
    "Tokyo";
    {
        "exemplarCity";
        "Tokio";
    }
```

```

    }
    "Tomsk";
    {
        "exemplarCity";
        "Tomsk";
    }
    "Ulaanbaatar";
    {
        "exemplarCity";
        "Ulaanbaatar";
    }
    "Urumqi";
    {
        "exemplarCity";
        "Ürümqi";
    }
    "Ust-Nera";
    {
        "exemplarCity";
        "Ust-Nera";
    }
    "Vientiane";
    {
        "exemplarCity";
        "Vientiane";
    }
    "Vladivostok";
    {
        "exemplarCity";
        "Wladiwostok";
    }
    "Yakutsk";
    {
        "exemplarCity";
        "Jakutsk";
    }
    "Yekaterinburg";
    {
        "exemplarCity";
        "Jekaterinburg";
    }
    "Yerevan";
    {
        "exemplarCity";
        "Eriwan";
    }
}
"Indian";
{
    "Antananarivo";
    {
        "exemplarCity";
        "Antananarivo";
    }
    "Chagos";
    {
        "exemplarCity";

```

```
        "Chagos";
    }
    "Christmas";
    {
        "exemplarCity";
        "Weihnachtsinsel";
    }
    "Cocos";
    {
        "exemplarCity";
        "Cocos";
    }
    "Comoro";
    {
        "exemplarCity";
        "Komoren";
    }
    "Kerguelen";
    {
        "exemplarCity";
        "Kerguelen";
    }
    "Mahe";
    {
        "exemplarCity";
        "Mahe";
    }
    "Maldives";
    {
        "exemplarCity";
        "Malediven";
    }
    "Mauritius";
    {
        "exemplarCity";
        "Mauritius";
    }
    "Mayotte";
    {
        "exemplarCity";
        "Mayotte";
    }
    "Reunion";
    {
        "exemplarCity";
        "Réunion";
    }
}
"Australia";
{
    "Adelaide";
    {
        "exemplarCity";
        "Adelaide";
    }
    "Brisbane";
    {
```

```
        "exemplarCity";
        "Brisbane";
    }
    "Broken_Hill";
    {
        "exemplarCity";
        "Broken Hill";
    }
    "Currie";
    {
        "exemplarCity";
        "Currie";
    }
    "Darwin";
    {
        "exemplarCity";
        "Darwin";
    }
    "Eucla";
    {
        "exemplarCity";
        "Eucla";
    }
    "Hobart";
    {
        "exemplarCity";
        "Hobart";
    }
    "Lindeman";
    {
        "exemplarCity";
        "Lindeman";
    }
    "Lord_Howe";
    {
        "exemplarCity";
        "Lord Howe";
    }
    "Melbourne";
    {
        "exemplarCity";
        "Melbourne";
    }
    "Perth";
    {
        "exemplarCity";
        "Perth";
    }
    "Sydney";
    {
        "exemplarCity";
        "Sydney";
    }
}
"Pacific";
{
    "Apia";
```

```
{
    "exemplarCity";
    "Apia";
}
"Auckland";
{
    "exemplarCity";
    "Auckland";
}
"Bougainville";
{
    "exemplarCity";
    "Bougainville";
}
"Chatham";
{
    "exemplarCity";
    "Chatham";
}
"Easter";
{
    "exemplarCity";
    "Osterinsel";
}
"Efate";
{
    "exemplarCity";
    "Efate";
}
"Enderbury";
{
    "exemplarCity";
    "Enderbury";
}
"Fakaofo";
{
    "exemplarCity";
    "Fakaofo";
}
"Fiji";
{
    "exemplarCity";
    "Fidschi";
}
"Funafuti";
{
    "exemplarCity";
    "Funafuti";
}
"Galapagos";
{
    "exemplarCity";
    "Galapagos";
}
"Gambier";
{
    "exemplarCity";
```

```
        "Gambier";
    }
    "Guadalcanal";
    {
        "exemplarCity";
        "Guadalcanal";
    }
    "Guam";
    {
        "exemplarCity";
        "Guam";
    }
    "Honolulu";
    {
        "exemplarCity";
        "Honolulu";
    }
    "Johnston";
    {
        "exemplarCity";
        "Johnston";
    }
    "Kiritimati";
    {
        "exemplarCity";
        "Kiritimati";
    }
    "Kosrae";
    {
        "exemplarCity";
        "Kosrae";
    }
    "Kwajalein";
    {
        "exemplarCity";
        "Kwajalein";
    }
    "Majuro";
    {
        "exemplarCity";
        "Majuro";
    }
    "Marquesas";
    {
        "exemplarCity";
        "Marquesas";
    }
    "Midway";
    {
        "exemplarCity";
        "Midway";
    }
    "Nauru";
    {
        "exemplarCity";
        "Nauru";
    }
}
```

```
"Niue";
{
  "exemplarCity";
  "Niue";
}
"Norfolk";
{
  "exemplarCity";
  "Norfolk";
}
"Noumea";
{
  "exemplarCity";
  "Noumea";
}
"Pago_Pago";
{
  "exemplarCity";
  "Pago Pago";
}
"Palau";
{
  "exemplarCity";
  "Palau";
}
"Pitcairn";
{
  "exemplarCity";
  "Pitcairn";
}
"Ponape";
{
  "exemplarCity";
  "Pohnpei";
}
"Port_Moresby";
{
  "exemplarCity";
  "Port Moresby";
}
"Rarotonga";
{
  "exemplarCity";
  "Rarotonga";
}
"Saipan";
{
  "exemplarCity";
  "Saipan";
}
"Tahiti";
{
  "exemplarCity";
  "Tahiti";
}
"Tarawa";
{
```

```
        "exemplarCity";
        "Tarawa";
    }
    "Tongatapu";
    {
        "exemplarCity";
        "Tongatapu";
    }
    "Truk";
    {
        "exemplarCity";
        "Chuuk";
    }
    "Wake";
    {
        "exemplarCity";
        "Wake";
    }
    "Wallis";
    {
        "exemplarCity";
        "Wallis";
    }
}
"Arctic";
{
    "Longyearbyen";
    {
        "exemplarCity";
        "Longyearbyen";
    }
}
"Antarctica";
{
    "Casey";
    {
        "exemplarCity";
        "Casey";
    }
    "Davis";
    {
        "exemplarCity";
        "Davis";
    }
    "DumontDUrville";
    {
        "exemplarCity";
        "Dumont d'Urville";
    }
    "Macquarie";
    {
        "exemplarCity";
        "Macquarie";
    }
    "Mawson";
    {
        "exemplarCity";
```



```
        "Mawson";
    }
    "McMurdo";
    {
        "exemplarCity";
        "McMurdo";
    }
    "Palmer";
    {
        "exemplarCity";
        "Palmer";
    }
    "Rothera";
    {
        "exemplarCity";
        "Rothera";
    }
    "Syowa";
    {
        "exemplarCity";
        "Syowa";
    }
    "Troll";
    {
        "exemplarCity";
        "Troll";
    }
    "Vostok";
    {
        "exemplarCity";
        "Wostok";
    }
}
"Etc";
{
    "GMT";
    {
        "exemplarCity";
        "GMT";
    }
    "GMT1";
    {
        "exemplarCity";
        "GMT+1";
    }
    "GMT10";
    {
        "exemplarCity";
        "GMT+10";
    }
    "GMT11";
    {
        "exemplarCity";
        "GMT+11";
    }
    "GMT12";
    {
```

```
        "exemplarCity";
        "GMT+12";
    }
    "GMT2";
    {
        "exemplarCity";
        "GMT+2";
    }
    "GMT3";
    {
        "exemplarCity";
        "GMT+3";
    }
    "GMT4";
    {
        "exemplarCity";
        "GMT+4";
    }
    "GMT5";
    {
        "exemplarCity";
        "GMT+5";
    }
    "GMT6";
    {
        "exemplarCity";
        "GMT+6";
    }
    "GMT7";
    {
        "exemplarCity";
        "GMT+7";
    }
    "GMT8";
    {
        "exemplarCity";
        "GMT+8";
    }
    "GMT9";
    {
        "exemplarCity";
        "GMT+9";
    }
    "GMT-1";
    {
        "exemplarCity";
        "GMT-1";
    }
    "GMT-10";
    {
        "exemplarCity";
        "GMT-10";
    }
    "GMT-11";
    {
        "exemplarCity";
        "GMT-11";
    }
```

```
}
"GMT-12";
{
  "exemplarCity";
  "GMT-12";
}
"GMT-13";
{
  "exemplarCity";
  "GMT-13";
}
"GMT-14";
{
  "exemplarCity";
  "GMT-14";
}
"GMT-2";
{
  "exemplarCity";
  "GMT-2";
}
"GMT-3";
{
  "exemplarCity";
  "GMT-3";
}
"GMT-4";
{
  "exemplarCity";
  "GMT-4";
}
"GMT-5";
{
  "exemplarCity";
  "GMT-5";
}
"GMT-6";
{
  "exemplarCity";
  "GMT-6";
}
"GMT-7";
{
  "exemplarCity";
  "GMT-7";
}
"GMT-8";
{
  "exemplarCity";
  "GMT-8";
}
"GMT-9";
{
  "exemplarCity";
  "GMT-9";
}
"Unknown";
```

```

        {
            "exemplarCity";
            "Unbekannt";
        }
    }
}
"metazone";
{
    "Acre";
    {
        "long";
        {
            "generic";
            "Acre-Zeit",
            "standard";
            "Acre-Normalzeit",
            "daylight";
            "Acre-Sommerzeit";
        }
    }
    "Afghanistan";
    {
        "long";
        {
            "standard";
            "Afghanistan-Zeit";
        }
    }
    "Africa_Central";
    {
        "long";
        {
            "standard";
            "Zentralafrikanische Zeit";
        }
    }
    "Africa_Eastern";
    {
        "long";
        {
            "standard";
            "Ostafrikanische Zeit";
        }
    }
    "Africa_Southern";
    {
        "long";
        {
            "standard";
            "Südafrikanische Zeit";
        }
    }
    "Africa_Western";
    {
        "long";
        {
            "generic";

```

```

        "Westafrikanische Zeit",
        "standard";
        "Westafrikanische Normalzeit",
        "daylight";
        "Westafrikanische Sommerzeit";
    }
}
"Alaska";
{
    "long";
    {
        "generic";
        "Alaska-Zeit",
        "standard";
        "Alaska-Normalzeit",
        "daylight";
        "Alaska-Sommerzeit";
    }
}
"Almaty";
{
    "long";
    {
        "generic";
        "Almaty-Zeit",
        "standard";
        "Almaty-Normalzeit",
        "daylight";
        "Almaty-Sommerzeit";
    }
}
"Amazon";
{
    "long";
    {
        "generic";
        "Amazonas-Zeit",
        "standard";
        "Amazonas-Normalzeit",
        "daylight";
        "Amazonas-Sommerzeit";
    }
}
"America_Central";
{
    "long";
    {
        "generic";
        "Nordamerikanische Inlandzeit",
        "standard";
        "Nordamerikanische Inland-Normalzeit",
        "daylight";
        "Nordamerikanische Inland-Sommerzeit";
    }
}
"America_Eastern";
{

```

```

        "long";
        {
            "generic";
            "Nordamerikanische Ostküstenzeit",
            "standard";
            "Nordamerikanische Ostküsten-Normalzeit",
            "daylight";
            "Nordamerikanische Ostküsten-Sommerzeit";
        }
    }
    "America_Mountain";
    {
        "long";
        {
            "generic";
            "Rocky-Mountain-Zeit",
            "standard";
            "Rocky Mountain-Normalzeit",
            "daylight";
            "Rocky-Mountain-Sommerzeit";
        }
    }
    "America_Pacific";
    {
        "long";
        {
            "generic";
            "Nordamerikanische Westküstenzeit",
            "standard";
            "Nordamerikanische Westküsten-Normalzeit",
            "daylight";
            "Nordamerikanische Westküsten-Sommerzeit";
        }
    }
    "Anadyr";
    {
        "long";
        {
            "generic";
            "Anadyr Zeit",
            "standard";
            "Anadyr Normalzeit",
            "daylight";
            "Anadyr Sommerzeit";
        }
    }
    "Apia";
    {
        "long";
        {
            "generic";
            "Apia-Zeit",
            "standard";
            "Apia-Normalzeit",
            "daylight";
            "Apia-Sommerzeit";
        }
    }

```

```

    }
    "Aqtau";
    {
        "long";
        {
            "generic";
            "Aqtau-Zeit",
            "standard";
            "Aqtau-Normalzeit",
            "daylight";
            "Aqtau-Sommerzeit";
        }
    }
    "Aqtobe";
    {
        "long";
        {
            "generic";
            "Aqtöbe-Zeit",
            "standard";
            "Aqtöbe-Normalzeit",
            "daylight";
            "Aqtöbe-Sommerzeit";
        }
    }
    "Arabian";
    {
        "long";
        {
            "generic";
            "Arabische Zeit",
            "standard";
            "Arabische Normalzeit",
            "daylight";
            "Arabische Sommerzeit";
        }
    }
    "Argentina";
    {
        "long";
        {
            "generic";
            "Argentinische Zeit",
            "standard";
            "Argentinische Normalzeit",
            "daylight";
            "Argentinische Sommerzeit";
        }
    }
    "Argentina_Western";
    {
        "long";
        {
            "generic";
            "Westargentinische Zeit",
            "standard";
            "Westargentinische Normalzeit",

```

```

        "daylight";
        "Westargentinische Sommerzeit";
    }
}
"Armenia";
{
    "long";
    {
        "generic";
        "Armenische Zeit",
        "standard";
        "Armenische Normalzeit",
        "daylight";
        "Armenische Sommerzeit";
    }
}
"Atlantic";
{
    "long";
    {
        "generic";
        "Atlantik-Zeit",
        "standard";
        "Atlantik-Normalzeit",
        "daylight";
        "Atlantik-Sommerzeit";
    }
}
"Australia_Central";
{
    "long";
    {
        "generic";
        "Zentralaustralische Zeit",
        "standard";
        "Zentralaustralische Normalzeit",
        "daylight";
        "Zentralaustralische Sommerzeit";
    }
}
"Australia_CentralWestern";
{
    "long";
    {
        "generic";
        "Zentral-/Westaustralische Zeit",
        "standard";
        "Zentral-/Westaustralische Normalzeit",
        "daylight";
        "Zentral-/Westaustralische Sommerzeit";
    }
}
"Australia_Eastern";
{
    "long";
    {
        "generic";

```



```

        "Ostaustralische Zeit",
        "standard";
        "Ostaustralische Normalzeit",
        "daylight";
        "Ostaustralische Sommerzeit";
    }
}
"Australia_Western";
{
    "long";
    {
        "generic";
        "Westaustralische Zeit",
        "standard";
        "Westaustralische Normalzeit",
        "daylight";
        "Westaustralische Sommerzeit";
    }
}
"Azerbaijan";
{
    "long";
    {
        "generic";
        "Aserbaidtschanische Zeit",
        "standard";
        "Aserbeidschanische Normalzeit",
        "daylight";
        "Aserbaidtschanische Sommerzeit";
    }
}
"Azores";
{
    "long";
    {
        "generic";
        "Azoren-Zeit",
        "standard";
        "Azoren-Normalzeit",
        "daylight";
        "Azoren-Sommerzeit";
    }
}
"Bangladesh";
{
    "long";
    {
        "generic";
        "Bangladesch-Zeit",
        "standard";
        "Bangladesch-Normalzeit",
        "daylight";
        "Bangladesch-Sommerzeit";
    }
}
"Bhutan";
{

```

```
        "long";
        {
            "standard";
            "Bhutan-Zeit";
        }
    }
    "Bolivia";
    {
        "long";
        {
            "standard";
            "Bolivianische Zeit";
        }
    }
    "Brasilia";
    {
        "long";
        {
            "generic";
            "Brasília-Zeit",
            "standard";
            "Brasília-Normalzeit",
            "daylight";
            "Brasília-Sommerzeit";
        }
    }
    "Brunei";
    {
        "long";
        {
            "standard";
            "Brunei-Zeit";
        }
    }
    "Cape_Verde";
    {
        "long";
        {
            "generic";
            "Cabo-Verde-Zeit",
            "standard";
            "Cabo-Verde-Normalzeit",
            "daylight";
            "Cabo-Verde-Sommerzeit";
        }
    }
    "Casey";
    {
        "long";
        {
            "standard";
            "Casey-Zeit";
        }
    }
    "Chamorro";
    {
        "long";
```

```
        {
            "standard";
            "Chamorro-Zeit";
        }
    }
    "Chatham";
    {
        "long";
        {
            "generic";
            "Chatham-Zeit",
            "standard";
            "Chatham-Normalzeit",
            "daylight";
            "Chatham-Sommerzeit";
        }
    }
    "Chile";
    {
        "long";
        {
            "generic";
            "Chilenische Zeit",
            "standard";
            "Chilenische Normalzeit",
            "daylight";
            "Chilenische Sommerzeit";
        }
    }
    "China";
    {
        "long";
        {
            "generic";
            "Chinesische Zeit",
            "standard";
            "Chinesische Normalzeit",
            "daylight";
            "Chinesische Sommerzeit";
        }
    }
    "Choibalsan";
    {
        "long";
        {
            "generic";
            "Tschoibalsan-Zeit",
            "standard";
            "Tschoibalsan-Normalzeit",
            "daylight";
            "Tschoibalsan-Sommerzeit";
        }
    }
    "Christmas";
    {
        "long";
        {
```

```

        "standard";
        "Weihnachtsinsel-Zeit";
    }
}
"Cocos";
{
    "long";
    {
        "standard";
        "Kokosinseln-Zeit";
    }
}
"Colombia";
{
    "long";
    {
        "generic";
        "Kolumbianische Zeit",
        "standard";
        "Kolumbianische Normalzeit",
        "daylight";
        "Kolumbianische Sommerzeit";
    }
}
"Cook";
{
    "long";
    {
        "generic";
        "Cookinseln-Zeit",
        "standard";
        "Cookinseln-Normalzeit",
        "daylight";
        "Cookinseln-Sommerzeit";
    }
}
"Cuba";
{
    "long";
    {
        "generic";
        "Kubanische Zeit",
        "standard";
        "Kubanische Normalzeit",
        "daylight";
        "Kubanische Sommerzeit";
    }
}
"Davis";
{
    "long";
    {
        "standard";
        "Davis-Zeit";
    }
}
"DumontDUrville";

```

```

    {
      "long";
      {
        "standard";
        "Dumont-d'Urville-Zeit";
      }
    }
    "East_Timor";
    {
      "long";
      {
        "standard";
        "Osttimor-Zeit";
      }
    }
    "Easter";
    {
      "long";
      {
        "generic";
        "Osterinsel-Zeit",
        "standard";
        "Osterinsel-Normalzeit",
        "daylight";
        "Osterinsel-Sommerzeit";
      }
    }
    "Ecuador";
    {
      "long";
      {
        "standard";
        "Ecuadorianische Zeit";
      }
    }
    "Europe_Central";
    {
      "long";
      {
        "generic";
        "Mitteleuropäische Zeit",
        "standard";
        "Mitteleuropäische Normalzeit",
        "daylight";
        "Mitteleuropäische Sommerzeit";
      }
      "short";
      {
        "generic";
        "MEZ",
        "standard";
        "MEZ",
        "daylight";
        "MESZ";
      }
    }
    "Europe_Eastern";

```

```

    {
      "long";
      {
        "generic";
        "Osteuropäische Zeit",
        "standard";
        "Osteuropäische Normalzeit",
        "daylight";
        "Osteuropäische Sommerzeit";
      }
      "short";
      {
        "generic";
        "OEZ",
        "standard";
        "OEZ",
        "daylight";
        "OESZ";
      }
    }
    "Europe_Further_Eastern";
    {
      "long";
      {
        "standard";
        "Kaliningrader Zeit";
      }
    }
    "Europe_Western";
    {
      "long";
      {
        "generic";
        "Westeuropäische Zeit",
        "standard";
        "Westeuropäische Normalzeit",
        "daylight";
        "Westeuropäische Sommerzeit";
      }
      "short";
      {
        "generic";
        "WEZ",
        "standard";
        "WEZ",
        "daylight";
        "WESZ";
      }
    }
    "Falkland";
    {
      "long";
      {
        "generic";
        "Falklandinseln-Zeit",
        "standard";
        "Falklandinseln-Normalzeit",

```

```

        "daylight";
        "Falklandinseln-Sommerzeit";
    }
}
"Fiji";
{
    "long";
    {
        "generic";
        "Fidschi-Zeit",
        "standard";
        "Fidschi-Normalzeit",
        "daylight";
        "Fidschi-Sommerzeit";
    }
}
"French_Guiana";
{
    "long";
    {
        "standard";
        "Französisch-Guayana-Zeit";
    }
}
"French_Southern";
{
    "long";
    {
        "standard";
        "Französische Süd- und Antarktisgebiete-
Zeit";
    }
}
"Galapagos";
{
    "long";
    {
        "standard";
        "Galapagos-Zeit";
    }
}
"Gambier";
{
    "long";
    {
        "standard";
        "Gambier-Zeit";
    }
}
"Georgia";
{
    "long";
    {
        "generic";
        "Georgische Zeit",
        "standard";
        "Georgische Normalzeit",

```

```

        "daylight";
        "Georgische Sommerzeit";
    }
}
"Gilbert_Islands";
{
    "long";
    {
        "standard";
        "Gilbert-Inseln-Zeit";
    }
}
"GMT";
{
    "long";
    {
        "standard";
        "Mittlere Greenwich-Zeit";
    }
}
"Greenland_Eastern";
{
    "long";
    {
        "generic";
        "Ostgrönland-Zeit",
        "standard";
        "Ostgrönland-Normalzeit",
        "daylight";
        "Ostgrönland-Sommerzeit";
    }
}
"Greenland_Western";
{
    "long";
    {
        "generic";
        "Westgrönland-Zeit",
        "standard";
        "Westgrönland-Normalzeit",
        "daylight";
        "Westgrönland-Sommerzeit";
    }
}
"Guam";
{
    "long";
    {
        "standard";
        "Guam-Zeit";
    }
}
"Gulf";
{
    "long";
    {
        "standard";
    }
}

```



```

        "Golf-Zeit";
    }
}
"Guyana";
{
    "long";
    {
        "standard";
        "Guyana-Zeit";
    }
}
"Hawaii_Aleutian";
{
    "long";
    {
        "generic";
        "Hawaii-Aleuten-Zeit",
        "standard";
        "Hawaii-Aleuten-Normalzeit",
        "daylight";
        "Hawaii-Aleuten-Sommerzeit";
    }
}
"Hong_Kong";
{
    "long";
    {
        "generic";
        "Hongkong-Zeit",
        "standard";
        "Hongkong-Normalzeit",
        "daylight";
        "Hongkong-Sommerzeit";
    }
}
"Hovd";
{
    "long";
    {
        "generic";
        "Chowd-Zeit",
        "standard";
        "Chowd-Normalzeit",
        "daylight";
        "Chowd-Sommerzeit";
    }
}
"India";
{
    "long";
    {
        "standard";
        "Indische Zeit";
    }
}
"Indian_Ocean";
{

```

```
        "long";
        {
            "standard";
            "Indischer Ozean-Zeit";
        }
    }
    "Indochina";
    {
        "long";
        {
            "standard";
            "Indochina-Zeit";
        }
    }
    "Indonesia_Central";
    {
        "long";
        {
            "standard";
            "Zentralindonesische Zeit";
        }
    }
    "Indonesia_Eastern";
    {
        "long";
        {
            "standard";
            "Ostindonesische Zeit";
        }
    }
    "Indonesia_Western";
    {
        "long";
        {
            "standard";
            "Westindonesische Zeit";
        }
    }
    "Iran";
    {
        "long";
        {
            "generic";
            "Iranische Zeit",
            "standard";
            "Iranische Normalzeit",
            "daylight";
            "Iranische Sommerzeit";
        }
    }
    "Irkutsk";
    {
        "long";
        {
            "generic";
            "Irkutsk-Zeit",
            "standard";
```

```

        "Irkutsk-Normalzeit",
        "daylight";
        "Irkutsk-Sommerzeit";
    }
}
"Israel";
{
    "long";
    {
        "generic";
        "Israelische Zeit",
        "standard";
        "Israelische Normalzeit",
        "daylight";
        "Israelische Sommerzeit";
    }
}
"Japan";
{
    "long";
    {
        "generic";
        "Japanische Zeit",
        "standard";
        "Japanische Normalzeit",
        "daylight";
        "Japanische Sommerzeit";
    }
}
"Kamchatka";
{
    "long";
    {
        "generic";
        "Kamtschatka-Zeit",
        "standard";
        "Kamtschatka-Normalzeit",
        "daylight";
        "Kamtschatka-Sommerzeit";
    }
}
"Kazakhstan_Eastern";
{
    "long";
    {
        "standard";
        "Ostkasachische Zeit";
    }
}
"Kazakhstan_Western";
{
    "long";
    {
        "standard";
        "Westkasachische Zeit";
    }
}
}

```

```
"Korea";
{
  "long";
  {
    "generic";
    "Koreanische Zeit",
    "standard";
    "Koreanische Normalzeit",
    "daylight";
    "Koreanische Sommerzeit";
  }
}
"Kosrae";
{
  "long";
  {
    "standard";
    "Kosrae-Zeit";
  }
}
"Krasnojarsk";
{
  "long";
  {
    "generic";
    "Krasnojarsk-Zeit",
    "standard";
    "Krasnojarsk-Normalzeit",
    "daylight";
    "Krasnojarsk-Sommerzeit";
  }
}
"Kyrgystan";
{
  "long";
  {
    "standard";
    "Kirgisistan-Zeit";
  }
}
"Lanka";
{
  "long";
  {
    "standard";
    "Sri-Lanka-Zeit";
  }
}
"Line_Islands";
{
  "long";
  {
    "standard";
    "Linieninseln-Zeit";
  }
}
"Lord_Howe";
```

```

    {
      "long";
      {
        "generic";
        "Lord-Howe-Zeit",
        "standard";
        "Lord-Howe-Normalzeit",
        "daylight";
        "Lord-Howe-Sommerzeit";
      }
    }
    "Macau";
    {
      "long";
      {
        "generic";
        "Macau-Zeit",
        "standard";
        "Macau-Normalzeit",
        "daylight";
        "Macau-Sommerzeit";
      }
    }
    "Macquarie";
    {
      "long";
      {
        "standard";
        "Macquarieinsel-Zeit";
      }
    }
    "Magadan";
    {
      "long";
      {
        "generic";
        "Magadan-Zeit",
        "standard";
        "Magadan-Normalzeit",
        "daylight";
        "Magadan-Sommerzeit";
      }
    }
    "Malaysia";
    {
      "long";
      {
        "standard";
        "Malaysische Zeit";
      }
    }
    "Maldives";
    {
      "long";
      {
        "standard";
        "Malediven-Zeit";
      }
    }
  }

```

```

    }
  }
  "Marquesas";
  {
    "long";
    {
      "standard";
      "Marquesas-Zeit";
    }
  }
  "Marshall_Islands";
  {
    "long";
    {
      "standard";
      "Marshallinseln-Zeit";
    }
  }
  "Mauritius";
  {
    "long";
    {
      "generic";
      "Mauritius-Zeit",
      "standard";
      "Mauritius-Normalzeit",
      "daylight";
      "Mauritius-Sommerzeit";
    }
  }
  "Mawson";
  {
    "long";
    {
      "standard";
      "Mawson-Zeit";
    }
  }
  "Mexico_Northwest";
  {
    "long";
    {
      "generic";
      "Mexiko Nordwestliche Zone-Zeit",
      "standard";
      "Mexiko Nordwestliche Zone-Normalzeit",
      "daylight";
      "Mexiko Nordwestliche Zone-Sommerzeit";
    }
  }
  "Mexico_Pacific";
  {
    "long";
    {
      "generic";
      "Mexiko Pazifikzone-Zeit",
      "standard";
    }
  }

```

```
        "Mexiko Pazifikzone-Normalzeit",
        "daylight";
        "Mexiko Pazifikzone-Sommerzeit";
    }
}
"Mongolia";
{
    "long";
    {
        "generic";
        "Ulaanbaatar-Zeit",
        "standard";
        "Ulaanbaatar-Normalzeit",
        "daylight";
        "Ulaanbaatar-Sommerzeit";
    }
}
"Moscow";
{
    "long";
    {
        "generic";
        "Moskauer Zeit",
        "standard";
        "Moskauer Normalzeit",
        "daylight";
        "Moskauer Sommerzeit";
    }
}
"Myanmar";
{
    "long";
    {
        "standard";
        "Myanmar-Zeit";
    }
}
"Nauru";
{
    "long";
    {
        "standard";
        "Nauru-Zeit";
    }
}
"Nepal";
{
    "long";
    {
        "standard";
        "Nepalesische Zeit";
    }
}
"New_Caledonia";
{
    "long";
    {
```

```

        "generic";
        "Neukaledonische Zeit",
        "standard";
        "Neukaledonische Normalzeit",
        "daylight";
        "Neukaledonische Sommerzeit";
    }
}
"New_Zealand";
{
    "long";
    {
        "generic";
        "Neuseeland-Zeit",
        "standard";
        "Neuseeland-Normalzeit",
        "daylight";
        "Neuseeland-Sommerzeit";
    }
}
"Newfoundland";
{
    "long";
    {
        "generic";
        "Neufundland-Zeit",
        "standard";
        "Neufundland-Normalzeit",
        "daylight";
        "Neufundland-Sommerzeit";
    }
}
"Niue";
{
    "long";
    {
        "standard";
        "Niue-Zeit";
    }
}
"Norfolk";
{
    "long";
    {
        "standard";
        "Norfolkinsel-Zeit";
    }
}
"Noronha";
{
    "long";
    {
        "generic";
        "Fernando de Noronha-Zeit",
        "standard";
        "Fernando de Noronha-Normalzeit",
        "daylight";
    }
}

```



```

        "Fernando de Noronha-Sommerzeit";
    }
}
"North_Mariana";
{
    "long";
    {
        "standard";
        "Nördliche-Marianen-Zeit";
    }
}
"Novosibirsk";
{
    "long";
    {
        "generic";
        "Nowosibirsk-Zeit",
        "standard";
        "Nowosibirsk-Normalzeit",
        "daylight";
        "Nowosibirsk-Sommerzeit";
    }
}
"Omsk";
{
    "long";
    {
        "generic";
        "Omsk-Zeit",
        "standard";
        "Omsk-Normalzeit",
        "daylight";
        "Omsk-Sommerzeit";
    }
}
"Pakistan";
{
    "long";
    {
        "generic";
        "Pakistanische Zeit",
        "standard";
        "Pakistanische Normalzeit",
        "daylight";
        "Pakistanische Sommerzeit";
    }
}
"Palau";
{
    "long";
    {
        "standard";
        "Palau-Zeit";
    }
}
"Papua_New_Guinea";
{

```

```
        "long";
        {
            "standard";
            "Papua-Neuguinea-Zeit";
        }
    }
    "Paraguay";
    {
        "long";
        {
            "generic";
            "Paraguayanische Zeit",
            "standard";
            "Paraguayanische Normalzeit",
            "daylight";
            "Paraguayanische Sommerzeit";
        }
    }
    "Peru";
    {
        "long";
        {
            "generic";
            "Peruanische Zeit",
            "standard";
            "Peruanische Normalzeit",
            "daylight";
            "Peruanische Sommerzeit";
        }
    }
    "Philippines";
    {
        "long";
        {
            "generic";
            "Philippinische Zeit",
            "standard";
            "Philippinische Normalzeit",
            "daylight";
            "Philippinische Sommerzeit";
        }
    }
    "Phoenix_Islands";
    {
        "long";
        {
            "standard";
            "Phoenixinseln-Zeit";
        }
    }
    "Pierre_Miquelon";
    {
        "long";
        {
            "generic";
            "Saint-Pierre-und-Miquelon-Zeit",
            "standard";
```

```
        "Saint-Pierre-und-Miquelon-Normalzeit",
        "daylight";
        "Saint-Pierre-und-Miquelon-Sommerzeit";
    }
}
"Pitcairn";
{
    "long";
    {
        "standard";
        "Pitcairninseeln-Zeit";
    }
}
"Ponape";
{
    "long";
    {
        "standard";
        "Ponape-Zeit";
    }
}
"Pyongyang";
{
    "long";
    {
        "standard";
        "Pjöngjang-Zeit";
    }
}
"Qyzylorda";
{
    "long";
    {
        "generic";
        "Quysylorda-Zeit",
        "standard";
        "Quysylorda-Normalzeit",
        "daylight";
        "Qysylorda-Sommerzeit";
    }
}
"Reunion";
{
    "long";
    {
        "standard";
        "Réunion-Zeit";
    }
}
"Rothera";
{
    "long";
    {
        "standard";
        "Rothera-Zeit";
    }
}
}
```

```
"Sakhalin";
{
  "long";
  {
    "generic";
    "Sachalin-Zeit",
    "standard";
    "Sachalin-Normalzeit",
    "daylight";
    "Sachalin-Sommerzeit";
  }
}
"Samara";
{
  "long";
  {
    "generic";
    "Samara-Zeit",
    "standard";
    "Samara-Normalzeit",
    "daylight";
    "Samara-Sommerzeit";
  }
}
"Samoa";
{
  "long";
  {
    "generic";
    "Samoa-Zeit",
    "standard";
    "Samoa-Normalzeit",
    "daylight";
    "Samoa-Sommerzeit";
  }
}
"Seychelles";
{
  "long";
  {
    "standard";
    "Seychellen-Zeit";
  }
}
"Singapore";
{
  "long";
  {
    "standard";
    "Singapur-Zeit";
  }
}
"Solomon";
{
  "long";
  {
    "standard";
```

```

        "Salomoninseln-Zeit";
    }
}
"South_Georgia";
{
    "long";
    {
        "standard";
        "Südgeorgische Zeit";
    }
}
"Suriname";
{
    "long";
    {
        "standard";
        "Suriname-Zeit";
    }
}
"Syowa";
{
    "long";
    {
        "standard";
        "Syowa-Zeit";
    }
}
"Tahiti";
{
    "long";
    {
        "standard";
        "Tahiti-Zeit";
    }
}
"Taipei";
{
    "long";
    {
        "generic";
        "Taipeh-Zeit",
        "standard";
        "Taipeh-Normalzeit",
        "daylight";
        "Taipeh-Sommerzeit";
    }
}
"Tajikistan";
{
    "long";
    {
        "standard";
        "Tadschikistan-Zeit";
    }
}
"Tokelau";
{

```

```
        "long";
        {
            "standard";
            "Tokelau-Zeit";
        }
    }
    "Tonga";
    {
        "long";
        {
            "generic";
            "Tonganische Zeit",
            "standard";
            "Tonganische Normalzeit",
            "daylight";
            "Tonganische Sommerzeit";
        }
    }
    "Truk";
    {
        "long";
        {
            "standard";
            "Chuuk-Zeit";
        }
    }
    "Turkmenistan";
    {
        "long";
        {
            "generic";
            "Turkmenistan-Zeit",
            "standard";
            "Turkmenistan-Normalzeit",
            "daylight";
            "Turkmenistan-Sommerzeit";
        }
    }
    "Tuvalu";
    {
        "long";
        {
            "standard";
            "Tuvalu-Zeit";
        }
    }
    "Uruguay";
    {
        "long";
        {
            "generic";
            "Uruguayische Zeit",
            "standard";
            "Uruguayische Normalzeit",
            "daylight";
            "Uruguayische Sommerzeit";
        }
    }
}
```

```
}
"Uzbekistan";
{
  "long";
  {
    "generic";
    "Uzbekistan-Zeit",
    "standard";
    "Uzbekistan-Normalzeit",
    "daylight";
    "Uzbekistan-Sommerzeit";
  }
}
"Vanuatu";
{
  "long";
  {
    "generic";
    "Vanuatu-Zeit",
    "standard";
    "Vanuatu-Normalzeit",
    "daylight";
    "Vanuatu-Sommerzeit";
  }
}
"Venezuela";
{
  "long";
  {
    "standard";
    "Venezuela-Zeit";
  }
}
"Vladivostok";
{
  "long";
  {
    "generic";
    "Wladiwostok-Zeit",
    "standard";
    "Wladiwostok-Normalzeit",
    "daylight";
    "Wladiwostok-Sommerzeit";
  }
}
"Volgograd";
{
  "long";
  {
    "generic";
    "Wolgograd-Zeit",
    "standard";
    "Wolgograd-Normalzeit",
    "daylight";
    "Wolgograd-Sommerzeit";
  }
}
}
```

```

        "Vostok";
    {
        "long";
        {
            "standard";
            "Wostok-Zeit";
        }
    }
    "Wake";
    {
        "long";
        {
            "standard";
            "Wake-Insel-Zeit";
        }
    }
    "Wallis";
    {
        "long";
        {
            "standard";
            "Wallis-und-Futuna-Zeit";
        }
    }
    "Yakutsk";
    {
        "long";
        {
            "generic";
            "Jakutsk-Zeit",
                "standard";
            "Jakutsk-Normalzeit",
                "daylight";
            "Jakutsk-Sommerzeit";
        }
    }
    "Yekaterinburg";
    {
        "long";
        {
            "generic";
            "Jekaterinburg-Zeit",
                "standard";
            "Jekaterinburg-Normalzeit",
                "daylight";
            "Jekaterinburg-Sommerzeit";
        }
    }
}
}
}
}
}
}
}
}
}
}

```

[Functional-component]

CA-GREGORIAN.JSON

```
{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 12879 $",
          "_cldrVersion": "30.0.3"
        },
        "language": "de"
      },
      "dates": {
        "calendars": {
          "gregorian": {
            "months": {
              "format": {
                "abbreviated": {
                  "1": "Jan.",
                  "2": "Feb.",
                  "3": "März",
                  "4": "Apr.",
                  "5": "Mai",
                  "6": "Juni",
                  "7": "Juli",
                  "8": "Aug.",
                  "9": "Sep.",
                  "10": "Okt.",
                  "11": "Nov.",
                  "12": "Dez."
                },
                "narrow": {
                  "1": "J",
                  "2": "F",
                  "3": "M",
                  "4": "A",
                  "5": "M",
                  "6": "J",
                  "7": "J",
                  "8": "A",
                  "9": "S",
                  "10": "O",
                  "11": "N",
                  "12": "D"
                },
                "wide": {
                  "1": "Januar",
                  "2": "Februar",
                  "3": "März",
                  "4": "April",
                  "5": "Mai",
                  "6": "Juni",
                  "7": "Juli",
                  "8": "August",
                  "9": "September",
                  "10": "Oktober",
                  "11": "November",

```

```

        "12": "Dezember"
      },
    },
    "stand-alone": {
      "abbreviated": {
        "1": "Jan",
        "2": "Feb",
        "3": "Mär",
        "4": "Apr",
        "5": "Mai",
        "6": "Jun",
        "7": "Jul",
        "8": "Aug",
        "9": "Sep",
        "10": "Okt",
        "11": "Nov",
        "12": "Dez"
      },
      "narrow": {
        "1": "J",
        "2": "F",
        "3": "M",
        "4": "A",
        "5": "M",
        "6": "J",
        "7": "J",
        "8": "A",
        "9": "S",
        "10": "O",
        "11": "N",
        "12": "D"
      },
      "wide": {
        "1": "Januar",
        "2": "Februar",
        "3": "März",
        "4": "April",
        "5": "Mai",
        "6": "Juni",
        "7": "Juli",
        "8": "August",
        "9": "September",
        "10": "Oktober",
        "11": "November",
        "12": "Dezember"
      }
    }
  },
  "days": {
    "format": {
      "abbreviated": {
        "sun": "So.",
        "mon": "Mo.",
        "tue": "Di.",
        "wed": "Mi.",
        "thu": "Do.",
        "fri": "Fr.",

```

```
        "sat": "Sa.",
      },
      "narrow": {
        "sun": "S",
        "mon": "M",
        "tue": "D",
        "wed": "M",
        "thu": "D",
        "fri": "F",
        "sat": "S"
      },
      "short": {
        "sun": "So.",
        "mon": "Mo.",
        "tue": "Di.",
        "wed": "Mi.",
        "thu": "Do.",
        "fri": "Fr.",
        "sat": "Sa."
      },
      "wide": {
        "sun": "Sonntag",
        "mon": "Montag",
        "tue": "Dienstag",
        "wed": "Mittwoch",
        "thu": "Donnerstag",
        "fri": "Freitag",
        "sat": "Samstag"
      }
    },
    "stand-alone": {
      "abbreviated": {
        "sun": "So",
        "mon": "Mo",
        "tue": "Di",
        "wed": "Mi",
        "thu": "Do",
        "fri": "Fr",
        "sat": "Sa"
      },
      "narrow": {
        "sun": "S",
        "mon": "M",
        "tue": "D",
        "wed": "M",
        "thu": "D",
        "fri": "F",
        "sat": "S"
      },
      "short": {
        "sun": "So.",
        "mon": "Mo.",
        "tue": "Di.",
        "wed": "Mi.",
        "thu": "Do.",
        "fri": "Fr.",
        "sat": "Sa."
      }
    }
  }
}
```

```

    },
    "wide": {
      "sun": "Sonntag",
      "mon": "Montag",
      "tue": "Dienstag",
      "wed": "Mittwoch",
      "thu": "Donnerstag",
      "fri": "Freitag",
      "sat": "Samstag"
    }
  },
  "quarters": {
    "format": {
      "abbreviated": {
        "1": "Q1",
        "2": "Q2",
        "3": "Q3",
        "4": "Q4"
      },
      "narrow": {
        "1": "1",
        "2": "2",
        "3": "3",
        "4": "4"
      },
      "wide": {
        "1": "1. Quartal",
        "2": "2. Quartal",
        "3": "3. Quartal",
        "4": "4. Quartal"
      }
    },
    "stand-alone": {
      "abbreviated": {
        "1": "Q1",
        "2": "Q2",
        "3": "Q3",
        "4": "Q4"
      },
      "narrow": {
        "1": "1",
        "2": "2",
        "3": "3",
        "4": "4"
      },
      "wide": {
        "1": "1. Quartal",
        "2": "2. Quartal",
        "3": "3. Quartal",
        "4": "4. Quartal"
      }
    }
  },
  "dayPeriods": {
    "format": {
      "abbreviated": {

```

```

        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "morgens",
        "morning2": "vormittags",
        "afternoon1": "mittags",
        "afternoon2": "nachmittags",
        "evening1": "abends",
        "night1": "nachts"
    },
    "narrow": {
        "midnight": "Mitternacht",
        "am": "vm.",
        "pm": "nm.",
        "morning1": "morgens",
        "morning2": "vormittags",
        "afternoon1": "mittags",
        "afternoon2": "nachmittags",
        "evening1": "abends",
        "night1": "nachts"
    },
    "wide": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "morgens",
        "morning2": "vormittags",
        "afternoon1": "mittags",
        "afternoon2": "nachmittags",
        "evening1": "abends",
        "night1": "nachts"
    }
},
"stand-alone": {
    "abbreviated": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "Morgen",
        "morning2": "Vormittag",
        "afternoon1": "Mittag",
        "afternoon2": "Nachmittag",
        "evening1": "Abend",
        "night1": "Nacht"
    },
    "narrow": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "Morgen",
        "morning2": "Vormittag",
        "afternoon1": "Mittag",
        "afternoon2": "Nachmittag",
        "evening1": "Abend",
        "night1": "Nacht"
    },
    "wide": {

```

```

        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "Morgen",
        "morning2": "Vormittag",
        "afternoon1": "Mittag",
        "afternoon2": "Nachmittag",
        "evening1": "Abend",
        "night1": "Nacht"
    }
}
},
"eras": {
    "eraNames": {
        "0": "v. Chr.",
        "0-alt-variant": "vor unserer Zeitrechnung",
        "1": "n. Chr.",
        "1-alt-variant": "unserer Zeitrechnung"
    },
    "eraAbbr": {
        "0": "v. Chr.",
        "0-alt-variant": "v. u. Z.",
        "1": "n. Chr.",
        "1-alt-variant": "u. Z."
    },
    "eraNarrow": {
        "0": "v. Chr.",
        "0-alt-variant": "v. u. Z.",
        "1": "n. Chr.",
        "1-alt-variant": "u. Z."
    }
},
"dateFormats": {
    "full": "EEEE, d. MMMM y",
    "long": "d. MMMM y",
    "medium": "dd.MM.y",
    "short": "dd.MM.yy"
},
"timeFormats": {
    "full": "HH:mm:ss zzzz",
    "long": "HH:mm:ss z",
    "medium": "HH:mm:ss",
    "short": "HH:mm"
},
"dateTimeFormats": {
    "full": "{1} 'um' {0}",
    "long": "{1} 'um' {0}",
    "medium": "{1}, {0}",
    "short": "{1}, {0}",
    "availableFormats": {
        "d": "d",
        "E": "ccc",
        "Ed": "E, d.",
        "Ehm": "E h:mm a",
        "EHm": "E, HH:mm",
        "Ehms": "E, h:mm:ss a",
        "EHms": "E, HH:mm:ss",
    }
}

```

```

"Gy": "y G",
"GyMMM": "MMM y G",
"GyMMMd": "d. MMM y G",
"GyMMMED": "E, d. MMM y G",
"h": "h 'Uhr' a",
"H": "HH 'Uhr'",
"hm": "h:mm a",
"Hm": "HH:mm",
"hms": "h:mm:ss a",
"Hms": "HH:mm:ss",
"hmsv": "h:mm:ss a v",
"Hmsv": "HH:mm:ss v",
"hmV": "h:mm a v",
"HmV": "HH:mm v",
"M": "L",
"Md": "d.M.",
"MEd": "E, d.M.",
"MMd": "d.MM.",
"MMdd": "dd.MM.",
"MMM": "LLL",
"MMMd": "d. MMM",
"MMMED": "E, d. MMM",
"MMMMd": "d. MMMM",
"MMMMEd": "E, d. MMMM",
"MMMMW": "'Woche' W 'im' MMM",
"MMMMW": "'Woche' W 'im' MMM",
"ms": "mm:ss",
"y": "y",
"yM": "M.y",
"yMd": "d.M.y",
"yMEd": "E, d.M.y",
"yMM": "MM.y",
"yMMdd": "dd.MM.y",
"yMMM": "MMM y",
"yMMMd": "d. MMM y",
"yMMMED": "E, d. MMM y",
"yMMMM": "MMMM y",
"yQQQ": "QQQ y",
"yQQQQ": "QQQQ y",
"yw": "'Woche' w 'des' 'Jahres' y",
"yw": "'Woche' w 'des' 'Jahres' y"
},
"appendItems": {
  "Day": "{0} ({2}: {1})",
  "Day-Of-Week": "{0} {1}",
  "Era": "{1} {0}",
  "Hour": "{0} ({2}: {1})",
  "Minute": "{0} ({2}: {1})",
  "Month": "{0} ({2}: {1})",
  "Quarter": "{0} ({2}: {1})",
  "Second": "{0} ({2}: {1})",
  "Timezone": "{0} {1}",
  "Week": "{0} ({2}: {1})",
  "Year": "{1} {0}"
},
"intervalFormats": {
  "intervalFormatFallback": "{0} - {1}",

```

```

    "d": {
      "d": "d.-d."
    },
    "h": {
      "a": "h 'Uhr' a - h 'Uhr' a",
      "h": "h - h 'Uhr' a"
    },
    "H": {
      "H": "HH-HH 'Uhr'"
    },
    "hm": {
      "a": "h:mm a - h:mm a",
      "h": "h:mm-h:mm a",
      "m": "h:mm-h:mm a"
    },
    "Hm": {
      "H": "HH:mm-HH:mm 'Uhr'",
      "m": "HH:mm-HH:mm 'Uhr'"
    },
    "hmv": {
      "a": "h:mm a - h:mm a v",
      "h": "h:mm-h:mm a v",
      "m": "h:mm-h:mm a v"
    },
    "Hmv": {
      "H": "HH:mm-HH:mm 'Uhr' v",
      "m": "HH:mm-HH:mm 'Uhr' v"
    },
    "hv": {
      "a": "h a - h a v",
      "h": "h-h a v"
    },
    "Hv": {
      "H": "HH-HH 'Uhr' v"
    },
    "M": {
      "M": "M.-M."
    },
    "Md": {
      "d": "dd.MM. - dd.MM.",
      "M": "dd.MM. - dd.MM."
    },
    "MEd": {
      "d": "E, dd.MM. - E, dd.MM.",
      "M": "E, dd.MM. - E, dd.MM."
    },
    "MMM": {
      "M": "MMM-MMM"
    },
    "MMMd": {
      "d": "d.-d. MMM",
      "M": "d. MMM - d. MMM"
    },
    "MMMEd": {
      "d": "E, d. - E, d. MMM",
      "M": "E, d. MMM - E, d. MMM"
    },
  },

```



```
    "version";
    {
      "_number";
      "$Revision: 12879 $",
      "_cldrVersion";
      "30.0.3";
    }
    "language";
    "de";
  }
  "dates";
  {
    "calendars";
    {
      "gregorian";
      {
        "months";
        {
          "format";
          {
            "abbreviated";
            {
              "1";
              "Jan.",
              "2";
              "Feb.",
              "3";
              "März",
              "4";
              "Apr.",
              "5";
              "Mai",
              "6";
              "Juni",
              "7";
              "Juli",
              "8";
              "Aug.",
              "9";
              "Sep.",
              "10";
              "Okt.",
              "11";
              "Nov.",
              "12";
              "Dez.";
            }
            "narrow";
            {
              "1";
              "J",
              "2";
              "F",
              "3";
              "M",
              "4";
              "A",
```

```
        "5";
        "M",
        "6";
        "J",
        "7";
        "J",
        "8";
        "A",
        "9";
        "S",
        "10";
        "O",
        "11";
        "N",
        "12";
        "D";
    }
    "wide";
    {
        "1";
        "Januar",
        "2";
        "Februar",
        "3";
        "März",
        "4";
        "April",
        "5";
        "Mai",
        "6";
        "Juni",
        "7";
        "Juli",
        "8";
        "August",
        "9";
        "September",
        "10";
        "Oktober",
        "11";
        "November",
        "12";
        "Dezember";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
        "Jan",
        "2";
        "Feb",
        "3";
        "Mär",
        "4";
        "Apr",
```

```
        "5";
        "Mai",
        "6";
        "Jun",
        "7";
        "Jul",
        "8";
        "Aug",
        "9";
        "Sep",
        "10";
        "Okt",
        "11";
        "Nov",
        "12";
        "Dez";
    }
    "narrow";
    {
        "1";
        "J",
        "2";
        "F",
        "3";
        "M",
        "4";
        "A",
        "5";
        "M",
        "6";
        "J",
        "7";
        "J",
        "8";
        "A",
        "9";
        "S",
        "10";
        "O",
        "11";
        "N",
        "12";
        "D";
    }
    "wide";
    {
        "1";
        "Januar",
        "2";
        "Februar",
        "3";
        "März",
        "4";
        "April",
        "5";
        "Mai",
        "6";
```

```

        "Juni",
        "7";
        "Juli",
        "8";
        "August",
        "9";
        "September",
        "10";
        "Oktober",
        "11";
        "November",
        "12";
        "Dezember";
    }
}
}
"days";
{
    "format";
    {
        "abbreviated";
        {
            "sun";
            "So.",
            "mon";
            "Mo.",
            "tue";
            "Di.",
            "wed";
            "Mi.",
            "thu";
            "Do.",
            "fri";
            "Fr.",
            "sat";
            "Sa.";
        }
        "narrow";
        {
            "sun";
            "S",
            "mon";
            "M",
            "tue";
            "D",
            "wed";
            "M",
            "thu";
            "D",
            "fri";
            "F",
            "sat";
            "S";
        }
        "short";
        {
            "sun";

```

```

        "So.",
        "mon";
    "Mo.",
    "tue";
    "Di.",
    "wed";
    "Mi.",
    "thu";
    "Do.",
    "fri";
    "Fr.",
    "sat";
    "Sa.";
}
"wide";
{
    "sun";
    "Sonntag",
    "mon";
    "Montag",
    "tue";
    "Dienstag",
    "wed";
    "Mittwoch",
    "thu";
    "Donnerstag",
    "fri";
    "Freitag",
    "sat";
    "Samstag";
}
}
"stand-alone";
{
    "abbreviated";
    {
        "sun";
        "So",
        "mon";
        "Mo",
        "tue";
        "Di",
        "wed";
        "Mi",
        "thu";
        "Do",
        "fri";
        "Fr",
        "sat";
        "Sa";
    }
    "narrow";
    {
        "sun";
        "S",
        "mon";
        "M",

```

```

        "tue";
        "D",
        "wed";
        "M",
        "thu";
        "D",
        "fri";
        "F",
        "sat";
        "S";
    }
    "short";
    {
        "sun";
        "So.",
        "mon";
        "Mo.",
        "tue";
        "Di.",
        "wed";
        "Mi.",
        "thu";
        "Do.",
        "fri";
        "Fr.",
        "sat";
        "Sa.";
    }
    "wide";
    {
        "sun";
        "Sonntag",
        "mon";
        "Montag",
        "tue";
        "Dienstag",
        "wed";
        "Mittwoch",
        "thu";
        "Donnerstag",
        "fri";
        "Freitag",
        "sat";
        "Samstag";
    }
}
"quarters";
{
    "format";
    {
        "abbreviated";
        {
            "1";
            "Q1",
            "2";
            "Q2",

```

```

        "3";
        "Q3",
        "4";
        "Q4";
    }
    "narrow";
    {
        "1";
        "1",
        "2";
        "2",
        "3";
        "3",
        "4";
        "4";
    }
    "wide";
    {
        "1";
        "1. Quartal",
        "2";
        "2. Quartal",
        "3";
        "3. Quartal",
        "4";
        "4. Quartal";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
        "Q1",
        "2";
        "Q2",
        "3";
        "Q3",
        "4";
        "Q4";
    }
    "narrow";
    {
        "1";
        "1",
        "2";
        "2",
        "3";
        "3",
        "4";
        "4";
    }
    "wide";
    {
        "1";
        "1. Quartal",
        "2";

```



```

        "2. Quartal",
        "3";
        "3. Quartal",
        "4";
        "4. Quartal";
    }
}
}
"dayPeriods";
{
    "format";
    {
        "abbreviated";
        {
            "midnight";
            "Mitternacht",
            "am";
            "vorm.",
            "pm";
            "nachm.",
            "morning1";
            "morgens",
            "morning2";
            "vormittags",
            "afternoon1";
            "mittags",
            "afternoon2";
            "nachmittags",
            "evening1";
            "abends",
            "night1";
            "nachts";
        }
        "narrow";
        {
            "midnight";
            "Mitternacht",
            "am";
            "vm.",
            "pm";
            "nm.",
            "morning1";
            "morgens",
            "morning2";
            "vormittags",
            "afternoon1";
            "mittags",
            "afternoon2";
            "nachmittags",
            "evening1";
            "abends",
            "night1";
            "nachts";
        }
        "wide";
        {
            "midnight";

```

```

        "Mitternacht",
        "am";
    "vorm.",
    "pm";
    "nachm.",
    "morning1";
    "morgens",
    "morning2";
    "vormittags",
    "afternoon1";
    "mittags",
    "afternoon2";
    "nachmittags",
    "evening1";
    "abends",
    "night1";
    "nachts";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vorm.",
        "pm";
        "nachm.",
        "morning1";
        "Morgen",
        "morning2";
        "Vormittag",
        "afternoon1";
        "Mittag",
        "afternoon2";
        "Nachmittag",
        "evening1";
        "Abend",
        "night1";
        "Nacht";
    }
}
"narrow";
{
    "midnight";
    "Mitternacht",
    "am";
    "vorm.",
    "pm";
    "nachm.",
    "morning1";
    "Morgen",
    "morning2";
    "Vormittag",
    "afternoon1";
    "Mittag",
    "afternoon2";
}

```

```

        "Nachmittag",
        "evening1";
        "Abend",
        "night1";
        "Nacht";
    }
    "wide";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vorm.",
        "pm";
        "nachm.",
        "morning1";
        "Morgen",
        "morning2";
        "Vormittag",
        "afternoon1";
        "Mittag",
        "afternoon2";
        "Nachmittag",
        "evening1";
        "Abend",
        "night1";
        "Nacht";
    }
}
"eras";
{
    "eraNames";
    {
        "0";
        "v. Chr.",
        "0-alt-variant";
        "vor unserer Zeitrechnung",
        "1";
        "n. Chr.",
        "1-alt-variant";
        "unserer Zeitrechnung";
    }
    "eraAbbr";
    {
        "0";
        "v. Chr.",
        "0-alt-variant";
        "v. u. Z.",
        "1";
        "n. Chr.",
        "1-alt-variant";
        "u. Z.";
    }
    "eraNarrow";
    {
        "0";
        "v. Chr.",

```

```

        "0-alt-variant";
        "v. u. Z.",
        "1";
        "n. Chr.",
        "1-alt-variant";
        "u. Z.";
    }
}
"dateFormats";
{
    "full";
    "EEEE, d. MMMM y",
    "long";
    "d. MMMM y",
    "medium";
    "dd.MM.y",
    "short";
    "dd.MM.yy";
}
"timeFormats";
{
    "full";
    "HH:mm:ss zzzz",
    "long";
    "HH:mm:ss z",
    "medium";
    "HH:mm:ss",
    "short";
    "HH:mm";
}
"dateTimeFormats";
{
    "full";
    "{1} 'um' {0}",
    "long";
    "{1} 'um' {0}",
    "medium";
    "{1}, {0}",
    "short";
    "{1}, {0}",
    "availableFormats";
    {
        "d";
        "d",
        "E";
        "ccc",
        "Ed";
        "E, d.",
        "Ehm";
        "E h:mm a",
        "EHm";
        "E, HH:mm",
        "Ehms";
        "E, h:mm:ss a",
        "EHms";
        "E, HH:mm:ss",
        "Gy";
    }
}

```

```

"y G",
  "GyMMM";
"MMM y G",
  "GyMMMd";
"d. MMM y G",
  "GyMMMED";
"E, d. MMM y G",
  "h";
"h 'Uhr' a",
  "H";
"HH 'Uhr' ",
  "hm";
"h:mm a",
  "Hm";
"HH:mm",
  "hms";
"h:mm:ss a",
  "Hms";
"HH:mm:ss",
  "hmsv";
"h:mm:ss a v",
  "Hmsv";
"HH:mm:ss v",
  "hmv";
"h:mm a v",
  "Hmv";
"HH:mm v",
  "M";
"L",
  "Md";
"d.M.",
  "MEd";
"E, d.M.",
  "MMd";
"d.MM.",
  "MMdd";
"dd.MM.",
  "MMM";
"LLL",
  "MMMd";
"d. MMM",
  "MMMED";
"E, d. MMM",
  "MMMMd";
"d. MMMM",
  "MMMMMED";
"E, d. MMMM",
  "MMMMW";
"'Woche' W 'im' MMM",
  "MMMMW";
"'Woche' W 'im' MMM",
  "ms";
"mm:ss",
  "y";
"y",
  "yM";
"M.y",

```

```

        "yMd";
        "d.M.y",
        "yMEd";
        "E, d.M.y",
        "yMM";
        "MM.y",
        "yMMdd";
        "dd.MM.y",
        "yMMM";
        "MMM y",
        "yMMMd";
        "d. MMM y",
        "yMMMEd";
        "E, d. MMM y",
        "yMMMM";
        "MMMM y",
        "yQQQ";
        "QQQ y",
        "yQQQQ";
        "QQQQ y",
        "yw";
        "'Woche' w 'des' 'Jahres' y",
        "yw";
        "'Woche' w 'des' 'Jahres' y";
    }
    "appendItems";
    {
        "Day";
        "{0} ({2}: {1})",
        "Day-Of-Week";
        "{0} {1}",
        "Era";
        "{1} {0}",
        "Hour";
        "{0} ({2}: {1})",
        "Minute";
        "{0} ({2}: {1})",
        "Month";
        "{0} ({2}: {1})",
        "Quarter";
        "{0} ({2}: {1})",
        "Second";
        "{0} ({2}: {1})",
        "Timezone";
        "{0} {1}",
        "Week";
        "{0} ({2}: {1})",
        "Year";
        "{1} {0}";
    }
    "intervalFormats";
    {
        "intervalFormatFallback";
        "{0} - {1}",
        "d";
        {
            "d";

```

```

        "d.-d.";
    }
    "h";
    {
        "a";
        "h 'Uhr' a - h 'Uhr' a",
        "h";
        "h - h 'Uhr' a";
    }
    "H";
    {
        "H";
        "HH-HH 'Uhr'";
    }
    "hm";
    {
        "a";
        "h:mm a - h:mm a",
        "h";
        "h:mm-h:mm a",
        "m";
        "h:mm-h:mm a";
    }
    "Hm";
    {
        "H";
        "HH:mm-HH:mm 'Uhr' ",
        "m";
        "HH:mm-HH:mm 'Uhr' ";
    }
    "hmv";
    {
        "a";
        "h:mm a - h:mm a v",
        "h";
        "h:mm-h:mm a v",
        "m";
        "h:mm-h:mm a v";
    }
    "Hmv";
    {
        "H";
        "HH:mm-HH:mm 'Uhr' v",
        "m";
        "HH:mm-HH:mm 'Uhr' v";
    }
    "hv";
    {
        "a";
        "h a - h a v",
        "h";
        "h-h a v";
    }
    "Hv";
    {
        "H";
        "HH-HH 'Uhr' v";
    }

```

```

    }
    "M";
    {
        "M";
        "M.-M.";
    }
    "Md";
    {
        "d";
        "dd.MM. - dd.MM.",
        "M";
        "dd.MM. - dd.MM.";
    }
    "MEd";
    {
        "d";
        "E, dd.MM. - E, dd.MM.",
        "M";
        "E, dd.MM. - E, dd.MM.";
    }
    "MMM";
    {
        "M";
        "MMM-MMM";
    }
    "MMMd";
    {
        "d";
        "d.-d. MMM",
        "M";
        "d. MMM - d. MMM";
    }
    "MMMEd";
    {
        "d";
        "E, d. - E, d. MMM",
        "M";
        "E, d. MMM - E, d. MMM";
    }
    "MMMM";
    {
        "M";
        "LLLL-LLLL";
    }
    "Y";
    {
        "Y";
        "Y-Y";
    }
    "yM";
    {
        "M";
        "MM.y - MM.y",
        "Y";
        "MM.y - MM.y";
    }
    "yMd";

```



```

{
    "d";
    "dd.MM.y - dd.MM.y",
        "M";
    "dd.MM.y - dd.MM.y",
        "y";
    "dd.MM.y - dd.MM.y";
}
"yMEd";
{
    "d";
    "E, dd.MM.y - E, dd.MM.y",
        "M";
    "E, dd.MM.y - E, dd.MM.y",
        "y";
    "E, dd.MM.y - E, dd.MM.y";
}
"yMMM";
{
    "M";
    "MMM-MMM y",
        "y";
    "MMM y - MMM y";
}
"yMMMd";
{
    "d";
    "d.-d. MMM y",
        "M";
    "d. MMM - d. MMM y",
        "y";
    "d. MMM y - d. MMM y";
}
"yMMMED";
{
    "d";
    "E, d. - E, d. MMM y",
        "M";
    "E, d. MMM - E, d. MMM y",
        "y";
    "E, d. MMM y - E, d. MMM y";
}
"yMMMM";
{
    "M";
    "MMMM-MMMM y",
        "y";
    "MMMM y - MMMM y";
}
}
}
}
}
}
}
}
}
}
}

```

CURRENCIES.JSON

```
{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31"
        },
        "language": "de"
      },
      "numbers": {
        "currencies": {
          "ADP": {
            "displayName": "Andorranische Pesete",
            "displayName-count-one": "Andorranische Pesete",
            "displayName-count-other": "Andorranische Peseten",
            "symbol": "ADP"
          },
          "AED": {
            "displayName": "VAE-Dirham",
            "displayName-count-one": "VAE-Dirham",
            "displayName-count-other": "VAE-Dirham",
            "symbol": "AED"
          },
          "AFA": {
            "displayName": "Afghanische Afghani (1927-2002)",
            "displayName-count-one": "Afghanische Afghani (1927-2002)",
            "displayName-count-other": "Afghanische Afghani (1927-2002)",
            "symbol": "AFA"
          },
          "AFN": {
            "displayName": "Afghanischer Afghani",
            "displayName-count-one": "Afghanischer Afghani",
            "displayName-count-other": "Afghanische Afghani",
            "symbol": "AFN"
          },
          "ALK": {
            "displayName": "Albanischer Lek (1946-1965)",
            "displayName-count-one": "Albanischer Lek (1946-1965)",
            "displayName-count-other": "Albanische Lek (1946-1965)"
          },
          "ALL": {
            "displayName": "Albanischer Lek",
            "displayName-count-one": "Albanischer Lek",
            "displayName-count-other": "Albanische Lek",
            "symbol": "ALL"
          },
          "AMD": {
            "displayName": "Armenischer Dram",
            "displayName-count-one": "Armenischer Dram",
            "displayName-count-other": "Armenische Dram",
            "symbol": "AMD"
          }
        }
      }
    }
  }
}
```

```

    "ANG": {
      "displayName": "Niederländische-Antillen-Gulden",
      "displayName-count-one": "Niederländische-Antillen-Gulden",
      "displayName-count-other": "Niederländische-Antillen-Gulden",
      "symbol": "ANG"
    },
    "AOA": {
      "displayName": "Angolanischer Kwanza",
      "displayName-count-one": "Angolanischer Kwanza",
      "displayName-count-other": "Angolanische Kwanza",
      "symbol": "AOA",
      "symbol-alt-narrow": "Kz"
    },
    "AOK": {
      "displayName": "Angolanischer Kwanza (1977-1990)",
      "displayName-count-one": "Angolanischer Kwanza (1977-1990)",
      "displayName-count-other": "Angolanische Kwanza (1977-1990)",
      "symbol": "AOK"
    },
    "AON": {
      "displayName": "Angolanischer Neuer Kwanza (1990-2000)",
      "displayName-count-one": "Angolanischer Neuer Kwanza (1990-2000)",
      "displayName-count-other": "Angolanische Neue Kwanza (1990-2000)",
      "symbol": "AON"
    },
    "AOR": {
      "displayName": "Angolanischer Kwanza Reajustado (1995-1999)",
      "displayName-count-one": "Angolanischer Kwanza Reajustado (1995-1999)",
      "displayName-count-other": "Angolanische Kwanza Reajustado (1995-1999)",
      "symbol": "AOR"
    },
    "ARA": {
      "displayName": "Argentinischer Austral",
      "displayName-count-one": "Argentinischer Austral",
      "displayName-count-other": "Argentinische Austral",
      "symbol": "ARA"
    },
    "ARL": {
      "displayName": "Argentinischer Peso Ley (1970-1983)",
      "displayName-count-one": "Argentinischer Peso Ley (1970-1983)",
      "displayName-count-other": "Argentinische Pesos Ley (1970-1983)",
      "symbol": "ARL"
    },
    "ARM": {
      "displayName": "Argentinischer Peso (1881-1970)",
      "displayName-count-one": "Argentinischer Peso (1881-1970)",
      "displayName-count-other": "Argentinische Pesos (1881-1970)",
      "symbol": "ARM"
    },
    "ARP": {
      "displayName": "Argentinischer Peso (1983-1985)",
      "displayName-count-one": "Argentinischer Peso (1983-1985)",
      "displayName-count-other": "Argentinische Peso (1983-1985)",

```

```

        "symbol": "ARP"
    },
    "ARS": {
        "displayName": "Argentinischer Peso",
        "displayName-count-one": "Argentinischer Peso",
        "displayName-count-other": "Argentinische Pesos",
        "symbol": "ARS",
        "symbol-alt-narrow": "$"
    },
    "ATS": {
        "displayName": "Österreichischer Schilling",
        "displayName-count-one": "Österreichischer Schilling",
        "displayName-count-other": "Österreichische Schilling",
        "symbol": "öS"
    },
    "AUD": {
        "displayName": "Australischer Dollar",
        "displayName-count-one": "Australischer Dollar",
        "displayName-count-other": "Australische Dollar",
        "symbol": "AU$",
        "symbol-alt-narrow": "$"
    },
    "AWG": {
        "displayName": "Aruba-Florin",
        "displayName-count-one": "Aruba-Florin",
        "displayName-count-other": "Aruba-Florin",
        "symbol": "AWG"
    },
    "AZM": {
        "displayName": "Aserbaidtschan-Manat (1993-2006)",
        "displayName-count-one": "Aserbaidtschan-Manat (1993-2006)",
        "displayName-count-other": "Aserbaidtschan-Manat (1993-2006)",
        "symbol": "AZM"
    },
    "AZN": {
        "displayName": "Aserbaidtschan-Manat",
        "displayName-count-one": "Aserbaidtschan-Manat",
        "displayName-count-other": "Aserbaidtschan-Manat",
        "symbol": "AZN"
    },
    "BAD": {
        "displayName": "Bosnien und Herzegowina Dinar (1992-1994)",
        "displayName-count-one": "Bosnien und Herzegowina Dinar (1992-1994)",
        "displayName-count-other": "Bosnien und Herzegowina Dinar (1992-1994)",
        "symbol": "BAD"
    },
    "BAM": {
        "displayName": "Bosnien und Herzegowina Konvertierbare Mark",
        "displayName-count-one": "Bosnien und Herzegowina Konvertierbare Mark",
        "displayName-count-other": "Bosnien und Herzegowina Konvertierbare Mark",
        "symbol": "BAM",
        "symbol-alt-narrow": "KM"
    },
    },

```

```

    "BAN": {
      "displayName": "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
      "displayName-count-one": "Bosnien und Herzegowina Neuer Dinar
(1994-1997)",
      "displayName-count-other": "Bosnien und Herzegowina Neue Dinar
(1994-1997)",
      "symbol": "BAN"
    },
    "BBD": {
      "displayName": "Barbados-Dollar",
      "displayName-count-one": "Barbados-Dollar",
      "displayName-count-other": "Barbados-Dollar",
      "symbol": "BBD",
      "symbol-alt-narrow": "$"
    },
    "BDT": {
      "displayName": "Bangladesch-Taka",
      "displayName-count-one": "Bangladesch-Taka",
      "displayName-count-other": "Bangladesch-Taka",
      "symbol": "BDT",
      "symbol-alt-narrow": "ট"
    },
    "BEC": {
      "displayName": "Belgischer Franc (konvertibel)",
      "displayName-count-one": "Belgischer Franc (konvertibel)",
      "displayName-count-other": "Belgische Franc (konvertibel)",
      "symbol": "BEC"
    },
    "BEF": {
      "displayName": "Belgischer Franc",
      "displayName-count-one": "Belgischer Franc",
      "displayName-count-other": "Belgische Franc",
      "symbol": "BEF"
    },
    "BEL": {
      "displayName": "Belgischer Finanz-Franc",
      "displayName-count-one": "Belgischer Finanz-Franc",
      "displayName-count-other": "Belgische Finanz-Franc",
      "symbol": "BEL"
    },
    "BGL": {
      "displayName": "Bulgarische Lew (1962-1999)",
      "displayName-count-one": "Bulgarische Lew (1962-1999)",
      "displayName-count-other": "Bulgarische Lew (1962-1999)",
      "symbol": "BGL"
    },
    "BGM": {
      "displayName": "Bulgarischer Lew (1952-1962)",
      "displayName-count-one": "Bulgarischer Lew (1952-1962)",
      "displayName-count-other": "Bulgarische Lew (1952-1962)",
      "symbol": "BGK"
    },
    "BGN": {
      "displayName": "Bulgarischer Lew",
      "displayName-count-one": "Bulgarischer Lew",
      "displayName-count-other": "Bulgarische Lew",

```

```

        "symbol": "BGN"
    },
    "BGO": {
        "displayName": "Bulgarischer Lew (1879-1952)",
        "displayName-count-one": "Bulgarischer Lew (1879-1952)",
        "displayName-count-other": "Bulgarische Lew (1879-1952)",
        "symbol": "BGJ"
    },
    "BHD": {
        "displayName": "Bahrain-Dinar",
        "displayName-count-one": "Bahrain-Dinar",
        "displayName-count-other": "Bahrain-Dinar",
        "symbol": "BHD"
    },
    "BIF": {
        "displayName": "Burundi-Franc",
        "displayName-count-one": "Burundi-Franc",
        "displayName-count-other": "Burundi-Francs",
        "symbol": "BIF"
    },
    "BMD": {
        "displayName": "Bermuda-Dollar",
        "displayName-count-one": "Bermuda-Dollar",
        "displayName-count-other": "Bermuda-Dollar",
        "symbol": "BMD",
        "symbol-alt-narrow": "$"
    },
    "BND": {
        "displayName": "Brunei-Dollar",
        "displayName-count-one": "Brunei-Dollar",
        "displayName-count-other": "Brunei-Dollar",
        "symbol": "BND",
        "symbol-alt-narrow": "$"
    },
    "BOB": {
        "displayName": "Bolivanischer Boliviano",
        "displayName-count-one": "Bolivanischer Boliviano",
        "displayName-count-other": "Bolivianische Bolivianos",
        "symbol": "BOB",
        "symbol-alt-narrow": "Bs"
    },
    "BOL": {
        "displayName": "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-one": "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-other": "Bolivianische Bolivianos (1863-1963)",
        "symbol": "BOL"
    },
    "BOP": {
        "displayName": "Bolivianischer Peso",
        "displayName-count-one": "Bolivianischer Peso",
        "displayName-count-other": "Bolivianische Peso",
        "symbol": "BOP"
    },
    "BOV": {
        "displayName": "Boliviansiche Mvdol",
        "displayName-count-one": "Boliviansiche Mvdol",

```

```

        "displayName-count-other": "Bolivianische Mvdol",
        "symbol": "BOV"
    },
    "BRB": {
        "displayName": "Brasilianischer Cruzeiro Novo (1967-1986)",
        "displayName-count-one": "Brasilianischer Cruzeiro Novo (1967-
1986)",
        "displayName-count-other": "Brasilianische Cruzeiro Novo (1967-
1986)",
        "symbol": "BRB"
    },
    "BRC": {
        "displayName": "Brasilianischer Cruzado (1986-1989)",
        "displayName-count-one": "Brasilianischer Cruzado (1986-1989)",
        "displayName-count-other": "Brasilianische Cruzado (1986-1989)",
        "symbol": "BRC"
    },
    "BRE": {
        "displayName": "Brasilianischer Cruzeiro (1990-1993)",
        "displayName-count-one": "Brasilianischer Cruzeiro (1990-1993)",
        "displayName-count-other": "Brasilianische Cruzeiro (1990-1993)",
        "symbol": "BRE"
    },
    "BRL": {
        "displayName": "Brasilianischer Real",
        "displayName-count-one": "Brasilianischer Real",
        "displayName-count-other": "Brasilianische Real",
        "symbol": "R$",
        "symbol-alt-narrow": "R$"
    },
    "BRN": {
        "displayName": "Brasilianischer Cruzado Novo (1989-1990)",
        "displayName-count-one": "Brasilianischer Cruzado Novo (1989-
1990)",
        "displayName-count-other": "Brasilianische Cruzado Novo (1989-
1990)",
        "symbol": "BRN"
    },
    "BRR": {
        "displayName": "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-one": "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-other": "Brasilianische Cruzeiro (1993-1994)",
        "symbol": "BRR"
    },
    "BRZ": {
        "displayName": "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-one": "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-other": "Brasilianischer Cruzeiro (1942-
1967)",
        "symbol": "BRZ"
    },
    "BSD": {
        "displayName": "Bahamas-Dollar",
        "displayName-count-one": "Bahamas-Dollar",
        "displayName-count-other": "Bahamas-Dollar",
        "symbol": "BSD",
        "symbol-alt-narrow": "$"
    }

```

```

    },
    "BTN": {
      "displayName": "Bhutan-Ngultrum",
      "displayName-count-one": "Bhutan-Ngultrum",
      "displayName-count-other": "Bhutan-Ngultrum",
      "symbol": "BTN"
    },
    "BUK": {
      "displayName": "Birmanischer Kyat",
      "displayName-count-one": "Birmanischer Kyat",
      "displayName-count-other": "Birmanische Kyat",
      "symbol": "BUK"
    },
    "BWP": {
      "displayName": "Botswanischer Pula",
      "displayName-count-one": "Botswanischer Pula",
      "displayName-count-other": "Botswanische Pula",
      "symbol": "BWP",
      "symbol-alt-narrow": "P"
    },
    "BYB": {
      "displayName": "Belarus-Rubel (1994-1999)",
      "displayName-count-one": "Belarus-Rubel (1994-1999)",
      "displayName-count-other": "Belarus-Rubel (1994-1999)",
      "symbol": "BYB"
    },
    "BYN": {
      "displayName": "Weißrussischer Rubel",
      "displayName-count-one": "Weißrussischer Rubel",
      "displayName-count-other": "Weißrussische Rubel",
      "symbol": "BYN",
      "symbol-alt-narrow": "p."
    },
    "BYR": {
      "displayName": "Weißrussischer Rubel (2000-2016)",
      "displayName-count-one": "Weißrussischer Rubel (2000-2016)",
      "displayName-count-other": "Weißrussische Rubel (2000-2016)",
      "symbol": "BYR"
    },
    "BZD": {
      "displayName": "Belize-Dollar",
      "displayName-count-one": "Belize-Dollar",
      "displayName-count-other": "Belize-Dollar",
      "symbol": "BZD",
      "symbol-alt-narrow": "$"
    },
    "CAD": {
      "displayName": "Kanadischer Dollar",
      "displayName-count-one": "Kanadischer Dollar",
      "displayName-count-other": "Kanadische Dollar",
      "symbol": "CA$",
      "symbol-alt-narrow": "$"
    },
    "CDF": {
      "displayName": "Kongo-Franc",
      "displayName-count-one": "Kongo-Franc",
      "displayName-count-other": "Kongo-Francs",

```



```

        "symbol": "CDF"
    },
    "CHE": {
        "displayName": "WIR-Euro",
        "displayName-count-one": "WIR-Euro",
        "displayName-count-other": "WIR-Euro",
        "symbol": "CHE"
    },
    "CHF": {
        "displayName": "Schweizer Franken",
        "displayName-count-one": "Schweizer Franken",
        "displayName-count-other": "Schweizer Franken",
        "symbol": "CHF"
    },
    "CHW": {
        "displayName": "WIR Franken",
        "displayName-count-one": "WIR Franken",
        "displayName-count-other": "WIR Franken",
        "symbol": "CHW"
    },
    "CLE": {
        "displayName": "Chilenischer Escudo",
        "displayName-count-one": "Chilenischer Escudo",
        "displayName-count-other": "Chilenische Escudo",
        "symbol": "CLE"
    },
    "CLF": {
        "displayName": "Chilenische Unidades de Fomento",
        "displayName-count-one": "Chilenische Unidades de Fomento",
        "displayName-count-other": "Chilenische Unidades de Fomento",
        "symbol": "CLF"
    },
    "CLP": {
        "displayName": "Chilenischer Peso",
        "displayName-count-one": "Chilenischer Peso",
        "displayName-count-other": "Chilenische Pesos",
        "symbol": "CLP",
        "symbol-alt-narrow": "$"
    },
    "CNX": {
        "displayName": "Dollar der Chinesischen Volksbank",
        "displayName-count-one": "Dollar der Chinesischen Volksbank",
        "displayName-count-other": "Dollar der Chinesischen Volksbank",
        "symbol": "CNX"
    },
    "CNY": {
        "displayName": "Renminbi Yuan",
        "displayName-count-one": "Chinesischer Yuan",
        "displayName-count-other": "Renminbi Yuan",
        "symbol": "CN¥",
        "symbol-alt-narrow": "¥"
    },
    "COP": {
        "displayName": "Kolumbianischer Peso",
        "displayName-count-one": "Kolumbianischer Peso",
        "displayName-count-other": "Kolumbianische Pesos",
        "symbol": "COP",

```

```

        "symbol-alt-narrow": "$"
    },
    "COU": {
        "displayName": "Kolumbianische Unidades de valor real",
        "displayName-count-one": "Kolumbianische Unidad de valor real",
        "displayName-count-other": "Kolumbianische Unidades de valor
real",
        "symbol": "COU"
    },
    "CRC": {
        "displayName": "Costa-Rica-Colón",
        "displayName-count-one": "Costa-Rica-Colón",
        "displayName-count-other": "Costa-Rica-Colón",
        "symbol": "CRC",
        "symbol-alt-narrow": "₡"
    },
    "CSD": {
        "displayName": "Serbischer Dinar (2002-2006)",
        "displayName-count-one": "Serbischer Dinar (2002-2006)",
        "displayName-count-other": "Serbische Dinar (2002-2006)",
        "symbol": "CSD"
    },
    "CSK": {
        "displayName": "Tschechoslowakische Krone",
        "displayName-count-one": "Tschechoslowakische Kronen",
        "displayName-count-other": "Tschechoslowakische Kronen",
        "symbol": "CSK"
    },
    "CUC": {
        "displayName": "Kubanischer Peso (konvertibel)",
        "displayName-count-one": "Kubanischer Peso (konvertibel)",
        "displayName-count-other": "Kubanische Pesos (konvertibel)",
        "symbol": "CUC",
        "symbol-alt-narrow": "Cub$"
    },
    "CUP": {
        "displayName": "Kubanischer Peso",
        "displayName-count-one": "Kubanischer Peso",
        "displayName-count-other": "Kubanische Pesos",
        "symbol": "CUP",
        "symbol-alt-narrow": "$"
    },
    "CVE": {
        "displayName": "Cabo-Verde-Escudo",
        "displayName-count-one": "Cabo-Verde-Escudo",
        "displayName-count-other": "Cabo-Verde-Escudos",
        "symbol": "CVE"
    },
    "CYP": {
        "displayName": "Zypern-Pfund",
        "displayName-count-one": "Zypern Pfund",
        "displayName-count-other": "Zypern Pfund",
        "symbol": "CYP"
    },
    "CZK": {
        "displayName": "Tschechische Krone",
        "displayName-count-one": "Tschechische Krone",

```

```

    "displayName-count-other": "Tschechische Kronen",
    "symbol": "CZK",
    "symbol-alt-narrow": "Kč"
  },
  "DDM": {
    "displayName": "Mark der DDR",
    "displayName-count-one": "Mark der DDR",
    "displayName-count-other": "Mark der DDR",
    "symbol": "DDM"
  },
  "DEM": {
    "displayName": "Deutsche Mark",
    "displayName-count-one": "Deutsche Mark",
    "displayName-count-other": "Deutsche Mark",
    "symbol": "DM"
  },
  "DJF": {
    "displayName": "Dschibuti-Franc",
    "displayName-count-one": "Dschibuti-Franc",
    "displayName-count-other": "Dschibuti-Franc",
    "symbol": "DJF"
  },
  "DKK": {
    "displayName": "Dänische Krone",
    "displayName-count-one": "Dänische Krone",
    "displayName-count-other": "Dänische Kronen",
    "symbol": "DKK",
    "symbol-alt-narrow": "kr"
  },
  "DOP": {
    "displayName": "Dominikanischer Peso",
    "displayName-count-one": "Dominikanischer Peso",
    "displayName-count-other": "Dominikanische Pesos",
    "symbol": "DOP",
    "symbol-alt-narrow": "$"
  },
  "DZD": {
    "displayName": "Algerischer Dinar",
    "displayName-count-one": "Algerischer Dinar",
    "displayName-count-other": "Algerische Dinar",
    "symbol": "DZD"
  },
  "ECS": {
    "displayName": "Ecuadorianischer Sucre",
    "displayName-count-one": "Ecuadorianischer Sucre",
    "displayName-count-other": "Ecuadorianische Sucre",
    "symbol": "ECS"
  },
  "ECV": {
    "displayName": "Verrechnungseinheit für Ecuador",
    "displayName-count-one": "Verrechnungseinheiten für Ecuador",
    "displayName-count-other": "Verrechnungseinheiten für Ecuador",
    "symbol": "ECV"
  },
  "EEK": {
    "displayName": "Estnische Krone",
    "displayName-count-one": "Estnische Krone",

```

```

    "displayName-count-other": "Estnische Kronen",
    "symbol": "EEK"
  },
  "EGP": {
    "displayName": "Ägyptisches Pfund",
    "displayName-count-one": "Ägyptisches Pfund",
    "displayName-count-other": "Ägyptische Pfund",
    "symbol": "EGP",
    "symbol-alt-narrow": "E£"
  },
  "ERN": {
    "displayName": "Eritreischer Nakfa",
    "displayName-count-one": "Eritreischer Nakfa",
    "displayName-count-other": "Eritreische Nakfa",
    "symbol": "ERN"
  },
  "ESA": {
    "displayName": "Spanische Peseta (A-Konten)",
    "displayName-count-one": "Spanische Peseta (A-Konten)",
    "displayName-count-other": "Spanische Peseten (A-Konten)",
    "symbol": "ESA"
  },
  "ESB": {
    "displayName": "Spanische Peseta (konvertibel)",
    "displayName-count-one": "Spanische Peseta (konvertibel)",
    "displayName-count-other": "Spanische Peseten (konvertibel)",
    "symbol": "ESB"
  },
  "ESP": {
    "displayName": "Spanische Peseta",
    "displayName-count-one": "Spanische Peseta",
    "displayName-count-other": "Spanische Peseten",
    "symbol": "ESP",
    "symbol-alt-narrow": "₧"
  },
  "ETB": {
    "displayName": "Äthiopischer Birr",
    "displayName-count-one": "Äthiopischer Birr",
    "displayName-count-other": "Äthiopische Birr",
    "symbol": "ETB"
  },
  "EUR": {
    "displayName": "Euro",
    "displayName-count-one": "Euro",
    "displayName-count-other": "Euro",
    "symbol": "€",
    "symbol-alt-narrow": "€"
  },
  "FIM": {
    "displayName": "Finnische Mark",
    "displayName-count-one": "Finnische Mark",
    "displayName-count-other": "Finnische Mark",
    "symbol": "FIM"
  },
  "FJD": {
    "displayName": "Fidschi-Dollar",
    "displayName-count-one": "Fidschi-Dollar",

```

```

        "displayName-count-other": "Fidschi-Dollar",
        "symbol": "FJD",
        "symbol-alt-narrow": "$"
    },
    "FKP": {
        "displayName": "Falkland-Pfund",
        "displayName-count-one": "Falkland-Pfund",
        "displayName-count-other": "Falkland-Pfund",
        "symbol": "FKP",
        "symbol-alt-narrow": "Fl£"
    },
    "FRF": {
        "displayName": "Französischer Franc",
        "displayName-count-one": "Französischer Franc",
        "displayName-count-other": "Französische Franc",
        "symbol": "FRF"
    },
    "GBP": {
        "displayName": "Britisches Pfund",
        "displayName-count-one": "Britisches Pfund",
        "displayName-count-other": "Britische Pfund",
        "symbol": "£",
        "symbol-alt-narrow": "£"
    },
    "GEK": {
        "displayName": "Georgischer Kupon Larit",
        "displayName-count-one": "Georgischer Kupon Larit",
        "displayName-count-other": "Georgische Kupon Larit",
        "symbol": "GEK"
    },
    "GEL": {
        "displayName": "Georgischer Lari",
        "displayName-count-one": "Georgischer Lari",
        "displayName-count-other": "Georgische Lari",
        "symbol": "GEL",
        "symbol-alt-narrow": "ლ",
        "symbol-alt-variant": "ლ"
    },
    "GHC": {
        "displayName": "Ghanaischer Cedi (1979–2007)",
        "displayName-count-one": "Ghanaischer Cedi (1979–2007)",
        "displayName-count-other": "Ghanaische Cedi (1979–2007)",
        "symbol": "GHC"
    },
    "GHS": {
        "displayName": "Ghanaischer Cedi",
        "displayName-count-one": "Ghanaischer Cedi",
        "displayName-count-other": "Ghanaische Cedi",
        "symbol": "GHS"
    },
    "GIP": {
        "displayName": "Gibraltar-Pfund",
        "displayName-count-one": "Gibraltar-Pfund",
        "displayName-count-other": "Gibraltar Pfund",
        "symbol": "GIP",
        "symbol-alt-narrow": "£"
    },
    },

```

```
"GMD": {
  "displayName": "Gambia-Dalasi",
  "displayName-count-one": "Gambia-Dalasi",
  "displayName-count-other": "Gambia-Dalasi",
  "symbol": "GMD"
},
"GNF": {
  "displayName": "Guinea-Franc",
  "displayName-count-one": "Guinea-Franc",
  "displayName-count-other": "Guinea-Franc",
  "symbol": "GNF",
  "symbol-alt-narrow": "F.G."
},
"GNS": {
  "displayName": "Guineischer Syli",
  "displayName-count-one": "Guineischer Syli",
  "displayName-count-other": "Guineische Syli",
  "symbol": "GNS"
},
"GQE": {
  "displayName": "Äquatorialguinea-Ekwele",
  "displayName-count-one": "Äquatorialguinea-Ekwele",
  "displayName-count-other": "Äquatorialguinea-Ekwele",
  "symbol": "GQE"
},
"GRD": {
  "displayName": "Griechische Drachme",
  "displayName-count-one": "Griechische Drachme",
  "displayName-count-other": "Griechische Drachmen",
  "symbol": "GRD"
},
"GTQ": {
  "displayName": "Guatemaltekinscher Quetzal",
  "displayName-count-one": "Guatemaltekinscher Quetzal",
  "displayName-count-other": "Guatemaltekinsche Quetzales",
  "symbol": "GTQ",
  "symbol-alt-narrow": "Q"
},
"GWE": {
  "displayName": "Portugiesisch Guinea Escudo",
  "displayName-count-one": "Portugiesisch Guinea Escudo",
  "displayName-count-other": "Portugiesisch Guinea Escudo",
  "symbol": "GWE"
},
"GWP": {
  "displayName": "Guinea-Bissau Peso",
  "displayName-count-one": "Guinea-Bissau Peso",
  "displayName-count-other": "Guinea-Bissau Pesos",
  "symbol": "GWP"
},
"GYD": {
  "displayName": "Guyana-Dollar",
  "displayName-count-one": "Guyana-Dollar",
  "displayName-count-other": "Guyana-Dollar",
  "symbol": "GYD",
  "symbol-alt-narrow": "$"
},
```

```
"HKD": {
  "displayName": "Hongkong-Dollar",
  "displayName-count-one": "Hongkong-Dollar",
  "displayName-count-other": "Hongkong-Dollar",
  "symbol": "HK$",
  "symbol-alt-narrow": "$"
},
"HNL": {
  "displayName": "Honduras-Lempira",
  "displayName-count-one": "Honduras-Lempira",
  "displayName-count-other": "Honduras-Lempira",
  "symbol": "HNL",
  "symbol-alt-narrow": "L"
},
"HRD": {
  "displayName": "Kroatischer Dinar",
  "displayName-count-one": "Kroatischer Dinar",
  "displayName-count-other": "Kroatische Dinar",
  "symbol": "HRD"
},
"HRK": {
  "displayName": "Kroatischer Kuna",
  "displayName-count-one": "Kroatischer Kuna",
  "displayName-count-other": "Kroatische Kuna",
  "symbol": "HRK",
  "symbol-alt-narrow": "kn"
},
"HTG": {
  "displayName": "Haitianische Gourde",
  "displayName-count-one": "Haitianische Gourde",
  "displayName-count-other": "Haitianische Gourdes",
  "symbol": "HTG"
},
"HUF": {
  "displayName": "Ungarischer Forint",
  "displayName-count-one": "Ungarischer Forint",
  "displayName-count-other": "Ungarische Forint",
  "symbol": "HUF",
  "symbol-alt-narrow": "Ft"
},
"IDR": {
  "displayName": "Indonesische Rupiah",
  "displayName-count-one": "Indonesische Rupiah",
  "displayName-count-other": "Indonesische Rupiah",
  "symbol": "IDR",
  "symbol-alt-narrow": "Rp"
},
"IEP": {
  "displayName": "Irishes Pfund",
  "displayName-count-one": "Irishes Pfund",
  "displayName-count-other": "Irische Pfund",
  "symbol": "IEP"
},
"ILP": {
  "displayName": "Israelisches Pfund",
  "displayName-count-one": "Israelisches Pfund",
  "displayName-count-other": "Israelische Pfund",
```

```

    "symbol": "ILP"
  },
  "ILR": {
    "displayName": "Israelischer Schekel (1980-1985)",
    "displayName-count-one": "Israelischer Schekel (1980-1985)",
    "displayName-count-other": "Israelische Schekel (1980-1985)"
  },
  "ILS": {
    "displayName": "Israelischer Neuer Schekel",
    "displayName-count-one": "Israelischer Neuer Schekel",
    "displayName-count-other": "Israelische Neue Schekel",
    "symbol": "₪",
    "symbol-alt-narrow": "₪"
  },
  "INR": {
    "displayName": "Indische Rupie",
    "displayName-count-one": "Indische Rupie",
    "displayName-count-other": "Indische Rupien",
    "symbol": "₹",
    "symbol-alt-narrow": "₹"
  },
  "IQD": {
    "displayName": "Irakischer Dinar",
    "displayName-count-one": "Irakischer Dinar",
    "displayName-count-other": "Irakische Dinar",
    "symbol": "IQD"
  },
  "IRR": {
    "displayName": "Iranischer Rial",
    "displayName-count-one": "Iranischer Rial",
    "displayName-count-other": "Iranische Rial",
    "symbol": "IRR"
  },
  "ISJ": {
    "displayName": "Isländische Krone (1918-1981)",
    "displayName-count-one": "Isländische Krone (1918-1981)",
    "displayName-count-other": "Isländische Kronen (1918-1981)"
  },
  "ISK": {
    "displayName": "Isländische Krone",
    "displayName-count-one": "Isländische Krone",
    "displayName-count-other": "Isländische Kronen",
    "symbol": "ISK",
    "symbol-alt-narrow": "kr"
  },
  "ITL": {
    "displayName": "Italienische Lira",
    "displayName-count-one": "Italienische Lira",
    "displayName-count-other": "Italienische Lire",
    "symbol": "ITL"
  },
  "JMD": {
    "displayName": "Jamaika-Dollar",
    "displayName-count-one": "Jamaika-Dollar",
    "displayName-count-other": "Jamaika-Dollar",
    "symbol": "JMD",
    "symbol-alt-narrow": "$"
  }

```



```

    },
    "JOD": {
      "displayName": "Jordanischer Dinar",
      "displayName-count-one": "Jordanischer Dinar",
      "displayName-count-other": "Jordanische Dinar",
      "symbol": "JOD"
    },
    "JPY": {
      "displayName": "Japanischer Yen",
      "displayName-count-one": "Japanischer Yen",
      "displayName-count-other": "Japanische Yen",
      "symbol": "¥",
      "symbol-alt-narrow": "¥"
    },
    "KES": {
      "displayName": "Kenia-Schilling",
      "displayName-count-one": "Kenia-Schilling",
      "displayName-count-other": "Kenia-Schilling",
      "symbol": "KES"
    },
    "KGS": {
      "displayName": "Kirgisischer Som",
      "displayName-count-one": "Kirgisischer Som",
      "displayName-count-other": "Kirgisische Som",
      "symbol": "KGS"
    },
    "KHR": {
      "displayName": "Kambodschanischer Riel",
      "displayName-count-one": "Kambodschanischer Riel",
      "displayName-count-other": "Kambodschanische Riel",
      "symbol": "KHR",
      "symbol-alt-narrow": "៛"
    },
    "KMF": {
      "displayName": "Komoren-Franc",
      "displayName-count-one": "Komoren-Franc",
      "displayName-count-other": "Komoren-Francs",
      "symbol": "KMF",
      "symbol-alt-narrow": "FC"
    },
    "KPW": {
      "displayName": "Nordkoreanischer Won",
      "displayName-count-one": "Nordkoreanischer Won",
      "displayName-count-other": "Nordkoreanische Won",
      "symbol": "KPW",
      "symbol-alt-narrow": "₩"
    },
    "KRH": {
      "displayName": "Südkoreanischer Hwan (1953-1962)",
      "displayName-count-one": "Südkoreanischer Hwan (1953-1962)",
      "displayName-count-other": "Südkoreanischer Hwan (1953-1962)",
      "symbol": "KRH"
    },
    "KRO": {
      "displayName": "Südkoreanischer Won (1945-1953)",
      "displayName-count-one": "Südkoreanischer Won (1945-1953)",

```

```

    "displayName-count-other": "Südkoreanischer Won (1945-1953)",
    "symbol": "KRO"
  },
  "KRW": {
    "displayName": "Südkoreanischer Won",
    "displayName-count-one": "Südkoreanischer Won",
    "displayName-count-other": "Südkoreanische Won",
    "symbol": "₩",
    "symbol-alt-narrow": "₩"
  },
  "KWD": {
    "displayName": "Kuwait-Dinar",
    "displayName-count-one": "Kuwait-Dinar",
    "displayName-count-other": "Kuwait-Dinar",
    "symbol": "KWD"
  },
  "KYD": {
    "displayName": "Kaiman-Dollar",
    "displayName-count-one": "Kaiman-Dollar",
    "displayName-count-other": "Kaiman-Dollar",
    "symbol": "KYD",
    "symbol-alt-narrow": "$"
  },
  "KZT": {
    "displayName": "Kasachischer Tenge",
    "displayName-count-one": "Kasachischer Tenge",
    "displayName-count-other": "Kasachische Tenge",
    "symbol": "KZT",
    "symbol-alt-narrow": "₸"
  },
  "LAK": {
    "displayName": "Laotischer Kip",
    "displayName-count-one": "Laotischer Kip",
    "displayName-count-other": "Laotische Kip",
    "symbol": "LAK",
    "symbol-alt-narrow": "₭"
  },
  "LBP": {
    "displayName": "Libanesisches Pfund",
    "displayName-count-one": "Libanesisches Pfund",
    "displayName-count-other": "Libanesische Pfund",
    "symbol": "LBP",
    "symbol-alt-narrow": "L£"
  },
  "LKR": {
    "displayName": "Sri-Lanka-Rupie",
    "displayName-count-one": "Sri-Lanka-Rupie",
    "displayName-count-other": "Sri-Lanka-Rupien",
    "symbol": "LKR",
    "symbol-alt-narrow": "Rs"
  },
  "LRD": {
    "displayName": "Liberianischer Dollar",
    "displayName-count-one": "Liberianischer Dollar",
    "displayName-count-other": "Liberianische Dollar",
    "symbol": "LRD",
    "symbol-alt-narrow": "$"
  }

```

```
    },
    "LSL": {
      "displayName": "Loti",
      "displayName-count-one": "Loti",
      "displayName-count-other": "Loti",
      "symbol": "LSL"
    },
    "LTL": {
      "displayName": "Litauischer Litas",
      "displayName-count-one": "Litauischer Litas",
      "displayName-count-other": "Litauische Litas",
      "symbol": "LTL",
      "symbol-alt-narrow": "Lt"
    },
    "LTT": {
      "displayName": "Litauischer Talonas",
      "displayName-count-one": "Litauische Talonas",
      "displayName-count-other": "Litauische Talonas",
      "symbol": "LTT"
    },
    "LUC": {
      "displayName": "Luxemburgischer Franc (konvertibel)",
      "displayName-count-one": "Luxemburgische Franc (konvertibel)",
      "displayName-count-other": "Luxemburgische Franc (konvertibel)",
      "symbol": "LUC"
    },
    "LUF": {
      "displayName": "Luxemburgischer Franc",
      "displayName-count-one": "Luxemburgische Franc",
      "displayName-count-other": "Luxemburgische Franc",
      "symbol": "LUF"
    },
    "LUL": {
      "displayName": "Luxemburgischer Finanz-Franc",
      "displayName-count-one": "Luxemburgische Finanz-Franc",
      "displayName-count-other": "Luxemburgische Finanz-Franc",
      "symbol": "LUL"
    },
    "LVL": {
      "displayName": "Lettischer Lats",
      "displayName-count-one": "Lettischer Lats",
      "displayName-count-other": "Lettische Lats",
      "symbol": "LVL",
      "symbol-alt-narrow": "Ls"
    },
    "LVR": {
      "displayName": "Lettischer Rubel",
      "displayName-count-one": "Lettische Rubel",
      "displayName-count-other": "Lettische Rubel",
      "symbol": "LVR"
    },
    "LYD": {
      "displayName": "Libyscher Dinar",
      "displayName-count-one": "Libyscher Dinar",
      "displayName-count-other": "Libysche Dinar",
      "symbol": "LYD"
    }
  },
```

```

"MAD": {
  "displayName": "Marokkanischer Dirham",
  "displayName-count-one": "Marokkanischer Dirham",
  "displayName-count-other": "Marokkanische Dirham",
  "symbol": "MAD"
},
"MAF": {
  "displayName": "Marokkanischer Franc",
  "displayName-count-one": "Marokkanische Franc",
  "displayName-count-other": "Marokkanische Franc",
  "symbol": "MAF"
},
"MCF": {
  "displayName": "Monegassischer Franc",
  "displayName-count-one": "Monegassischer Franc",
  "displayName-count-other": "Monegassische Franc",
  "symbol": "MCF"
},
"MDC": {
  "displayName": "Moldau-Cupon",
  "displayName-count-one": "Moldau-Cupon",
  "displayName-count-other": "Moldau-Cupon",
  "symbol": "MDC"
},
"MDL": {
  "displayName": "Moldau-Leu",
  "displayName-count-one": "Moldau-Leu",
  "displayName-count-other": "Moldau-Leu",
  "symbol": "MDL"
},
"MGA": {
  "displayName": "Madagaskar-Ariary",
  "displayName-count-one": "Madagaskar-Ariary",
  "displayName-count-other": "Madagaskar-Ariary",
  "symbol": "MGA",
  "symbol-alt-narrow": "Ar"
},
"MGF": {
  "displayName": "Madagaskar-Franc",
  "displayName-count-one": "Madagaskar-Franc",
  "displayName-count-other": "Madagaskar-Franc",
  "symbol": "MGF"
},
"MKD": {
  "displayName": "Mazedonischer Denar",
  "displayName-count-one": "Mazedonischer Denar",
  "displayName-count-other": "Mazedonische Denari",
  "symbol": "MKD"
},
"MKN": {
  "displayName": "Mazedonischer Denar (1992-1993)",
  "displayName-count-one": "Mazedonischer Denar (1992-1993)",
  "displayName-count-other": "Mazedonische Denar (1992-1993)",
  "symbol": "MKN"
},
"MLF": {
  "displayName": "Malischer Franc",

```

```

    "displayName-count-one": "Malische Franc",
    "displayName-count-other": "Malische Franc",
    "symbol": "MLF"
  },
  "MMK": {
    "displayName": "Myanmarischer Kyat",
    "displayName-count-one": "Myanmarischer Kyat",
    "displayName-count-other": "Myanmarische Kyat",
    "symbol": "MMK",
    "symbol-alt-narrow": "K"
  },
  "MNT": {
    "displayName": "Mongolischer Tögrög",
    "displayName-count-one": "Mongolischer Tögrög",
    "displayName-count-other": "Mongolische Tögrög",
    "symbol": "MNT",
    "symbol-alt-narrow": "₮"
  },
  "MOP": {
    "displayName": "Macao-Pataca",
    "displayName-count-one": "Macao-Pataca",
    "displayName-count-other": "Macao-Pataca",
    "symbol": "MOP"
  },
  "MRO": {
    "displayName": "Mauretanischer Ouguiya",
    "displayName-count-one": "Mauretanischer Ouguiya",
    "displayName-count-other": "Mauretanische Ouguiya",
    "symbol": "MRO"
  },
  "MTL": {
    "displayName": "Maltesische Lira",
    "displayName-count-one": "Maltesische Lira",
    "displayName-count-other": "Maltesische Lira",
    "symbol": "MTL"
  },
  "MTP": {
    "displayName": "Maltesisches Pfund",
    "displayName-count-one": "Maltesische Pfund",
    "displayName-count-other": "Maltesische Pfund",
    "symbol": "MTP"
  },
  "MUR": {
    "displayName": "Mauritius-Rupie",
    "displayName-count-one": "Mauritius-Rupie",
    "displayName-count-other": "Mauritius-Rupien",
    "symbol": "MUR",
    "symbol-alt-narrow": "Rs"
  },
  "MVP": {
    "displayName": "Malediven-Rupie (alt)",
    "displayName-count-one": "Malediven-Rupie (alt)",
    "displayName-count-other": "Malediven-Rupien (alt)"
  },
  "MVR": {
    "displayName": "Malediven-Rufiyaa",
    "displayName-count-one": "Malediven-Rufiyaa",

```

```

        "displayName-count-other": "Malediven-Rupien",
        "symbol": "MVR"
    },
    "MWK": {
        "displayName": "Malawi-Kwacha",
        "displayName-count-one": "Malawi-Kwacha",
        "displayName-count-other": "Malawi-Kwacha",
        "symbol": "MWK"
    },
    "MXN": {
        "displayName": "Mexikanischer Peso",
        "displayName-count-one": "Mexikanischer Peso",
        "displayName-count-other": "Mexikanische Pesos",
        "symbol": "MX$",
        "symbol-alt-narrow": "$"
    },
    "MXP": {
        "displayName": "Mexikanischer Silber-Peso (1861-1992)",
        "displayName-count-one": "Mexikanische Silber-Peso (1861-1992)",
        "displayName-count-other": "Mexikanische Silber-Pesos (1861-
1992)",
        "symbol": "MXP"
    },
    "MXV": {
        "displayName": "Mexicanischer Unidad de Inversion (UDI)",
        "displayName-count-one": "Mexicanischer Unidad de Inversion
(UDI)",
        "displayName-count-other": "Mexikanische Unidad de Inversion
(UDI)",
        "symbol": "MXV"
    },
    "MYR": {
        "displayName": "Malaysischer Ringgit",
        "displayName-count-one": "Malaysischer Ringgit",
        "displayName-count-other": "Malaysische Ringgit",
        "symbol": "MYR",
        "symbol-alt-narrow": "RM"
    },
    "MZE": {
        "displayName": "Mosambikanischer Escudo",
        "displayName-count-one": "Mozambikanische Escudo",
        "displayName-count-other": "Mozambikanische Escudo",
        "symbol": "MZE"
    },
    "MZM": {
        "displayName": "Mosambikanischer Metical (1980-2006)",
        "displayName-count-one": "Mosambikanischer Metical (1980-2006)",
        "displayName-count-other": "Mosambikanische Meticais (1980-
2006)",
        "symbol": "MZM"
    },
    "MZN": {
        "displayName": "Mosambikanischer Metical",
        "displayName-count-one": "Mosambikanischer Metical",
        "displayName-count-other": "Mosambikanische Meticais",
        "symbol": "MZN"
    },
    },

```

```

"NAD": {
  "displayName": "Namibia-Dollar",
  "displayName-count-one": "Namibia-Dollar",
  "displayName-count-other": "Namibia-Dollar",
  "symbol": "NAD",
  "symbol-alt-narrow": "$"
},
"NGN": {
  "displayName": "Nigerianischer Naira",
  "displayName-count-one": "Nigerianischer Naira",
  "displayName-count-other": "Nigerianische Naira",
  "symbol": "NGN",
  "symbol-alt-narrow": "₦"
},
"NIC": {
  "displayName": "Nicaraguanischer Córdoba (1988-1991)",
  "displayName-count-one": "Nicaraguanischer Córdoba (1988-1991)",
  "displayName-count-other": "Nicaraguanische Córdoba (1988-1991)",
  "symbol": "NIC"
},
"NIO": {
  "displayName": "Nicaragua-Córdoba",
  "displayName-count-one": "Nicaragua-Córdoba",
  "displayName-count-other": "Nicaragua-Córdobas",
  "symbol": "NIO",
  "symbol-alt-narrow": "C$"
},
"NLG": {
  "displayName": "Niederländischer Gulden",
  "displayName-count-one": "Niederländischer Gulden",
  "displayName-count-other": "Niederländische Gulden",
  "symbol": "NLG"
},
"NOK": {
  "displayName": "Norwegische Krone",
  "displayName-count-one": "Norwegische Krone",
  "displayName-count-other": "Norwegische Kronen",
  "symbol": "NOK",
  "symbol-alt-narrow": "kr"
},
"NPR": {
  "displayName": "Nepalesische Rupie",
  "displayName-count-one": "Nepalesische Rupie",
  "displayName-count-other": "Nepalesische Rupien",
  "symbol": "NPR",
  "symbol-alt-narrow": "Rs"
},
"NZD": {
  "displayName": "Neuseeland-Dollar",
  "displayName-count-one": "Neuseeland-Dollar",
  "displayName-count-other": "Neuseeland-Dollar",
  "symbol": "NZ$",
  "symbol-alt-narrow": "$"
},
"OMR": {
  "displayName": "Omanischer Rial",
  "displayName-count-one": "Omanischer Rial",

```

```

        "displayName-count-other": "Omanische Rials",
        "symbol": "OMR"
    },
    "PAB": {
        "displayName": "Panamaischer Balboa",
        "displayName-count-one": "Panamaischer Balboa",
        "displayName-count-other": "Panamaische Balboas",
        "symbol": "PAB"
    },
    "PEI": {
        "displayName": "Peruanischer Inti",
        "displayName-count-one": "Peruanische Inti",
        "displayName-count-other": "Peruanische Inti",
        "symbol": "PEI"
    },
    "PEN": {
        "displayName": "Peruanischer Sol",
        "displayName-count-one": "Peruanischer Sol",
        "displayName-count-other": "Peruanische Sol",
        "symbol": "PEN"
    },
    "PES": {
        "displayName": "Peruanischer Sol (1863-1965)",
        "displayName-count-one": "Peruanischer Sol (1863-1965)",
        "displayName-count-other": "Peruanische Sol (1863-1965)",
        "symbol": "PES"
    },
    "PGK": {
        "displayName": "Papua-Neuguineischer Kina",
        "displayName-count-one": "Papua-Neuguineischer Kina",
        "displayName-count-other": "Papua-Neuguineische Kina",
        "symbol": "PGK"
    },
    "PHP": {
        "displayName": "Philippinischer Peso",
        "displayName-count-one": "Philippinischer Peso",
        "displayName-count-other": "Philippinische Pesos",
        "symbol": "PHP",
        "symbol-alt-narrow": "₱"
    },
    "PKR": {
        "displayName": "Pakistanische Rupie",
        "displayName-count-one": "Pakistanische Rupie",
        "displayName-count-other": "Pakistanische Rupien",
        "symbol": "PKR",
        "symbol-alt-narrow": "Rs"
    },
    "PLN": {
        "displayName": "Polnischer Złoty",
        "displayName-count-one": "Polnischer Złoty",
        "displayName-count-other": "Polnische Złoty",
        "symbol": "PLN",
        "symbol-alt-narrow": "zł"
    },
    "PLZ": {
        "displayName": "Polnischer Zloty (1950-1995)",
        "displayName-count-one": "Polnischer Zloty (1950-1995)",

```



```

    "displayName-count-other": "Polnische Zloty (1950-1995)",
    "symbol": "PLZ"
  },
  "PTE": {
    "displayName": "Portugiesischer Escudo",
    "displayName-count-one": "Portugiesische Escudo",
    "displayName-count-other": "Portugiesische Escudo",
    "symbol": "PTE"
  },
  "PYG": {
    "displayName": "Paraguayischer Guaraní",
    "displayName-count-one": "Paraguayischer Guaraní",
    "displayName-count-other": "Paraguayische Guaraníes",
    "symbol": "PYG",
    "symbol-alt-narrow": "₲"
  },
  "QAR": {
    "displayName": "Katar-Riyal",
    "displayName-count-one": "Katar-Riyal",
    "displayName-count-other": "Katar-Riyal",
    "symbol": "QAR"
  },
  "RHD": {
    "displayName": "Rhodesischer Dollar",
    "displayName-count-one": "Rhodesische Dollar",
    "displayName-count-other": "Rhodesische Dollar",
    "symbol": "RHD"
  },
  "ROL": {
    "displayName": "Rumänischer Leu (1952-2006)",
    "displayName-count-one": "Rumänischer Leu (1952-2006)",
    "displayName-count-other": "Rumänische Leu (1952-2006)",
    "symbol": "ROL"
  },
  "RON": {
    "displayName": "Rumänischer Leu",
    "displayName-count-one": "Rumänischer Leu",
    "displayName-count-other": "Rumänische Leu",
    "symbol": "RON",
    "symbol-alt-narrow": "L"
  },
  "RSD": {
    "displayName": "Serbischer Dinar",
    "displayName-count-one": "Serbischer Dinar",
    "displayName-count-other": "Serbische Dinaren",
    "symbol": "RSD"
  },
  "RUB": {
    "displayName": "Russischer Rubel",
    "displayName-count-one": "Russischer Rubel",
    "displayName-count-other": "Russische Rubel",
    "symbol": "RUB",
    "symbol-alt-narrow": "₽"
  },
  "RUR": {
    "displayName": "Russischer Rubel (1991-1998)",
    "displayName-count-one": "Russischer Rubel (1991-1998)",

```

```

        "displayName-count-other": "Russische Rubel (1991-1998)",
        "symbol": "RUR",
        "symbol-alt-narrow": "p."
    },
    "RWF": {
        "displayName": "Ruanda-Franc",
        "displayName-count-one": "Ruanda-Franc",
        "displayName-count-other": "Ruanda-Francs",
        "symbol": "RWF",
        "symbol-alt-narrow": "F.Rw"
    },
    "SAR": {
        "displayName": "Saudi-Rial",
        "displayName-count-one": "Saudi-Rial",
        "displayName-count-other": "Saudi-Rial",
        "symbol": "SAR"
    },
    "SBD": {
        "displayName": "Salomonen-Dollar",
        "displayName-count-one": "Salomonen-Dollar",
        "displayName-count-other": "Salomonen-Dollar",
        "symbol": "SBD",
        "symbol-alt-narrow": "$"
    },
    "SCR": {
        "displayName": "Seychellen-Rupie",
        "displayName-count-one": "Seychellen-Rupie",
        "displayName-count-other": "Seychellen-Rupien",
        "symbol": "SCR"
    },
    "SDD": {
        "displayName": "Sudanesischer Dinar (1992-2007)",
        "displayName-count-one": "Sudanesischer Dinar (1992-2007)",
        "displayName-count-other": "Sudanesische Dinar (1992-2007)",
        "symbol": "SDD"
    },
    "SDG": {
        "displayName": "Sudanesisches Pfund",
        "displayName-count-one": "Sudanesisches Pfund",
        "displayName-count-other": "Sudanesische Pfund",
        "symbol": "SDG"
    },
    "SDP": {
        "displayName": "Sudanesisches Pfund (1957-1998)",
        "displayName-count-one": "Sudanesisches Pfund (1957-1998)",
        "displayName-count-other": "Sudanesische Pfund (1957-1998)",
        "symbol": "SDP"
    },
    "SEK": {
        "displayName": "Schwedische Krone",
        "displayName-count-one": "Schwedische Krone",
        "displayName-count-other": "Schwedische Kronen",
        "symbol": "SEK",
        "symbol-alt-narrow": "kr"
    },
    "SGD": {
        "displayName": "Singapur-Dollar",

```

```

        "displayName-count-one": "Singapur-Dollar",
        "displayName-count-other": "Singapur-Dollar",
        "symbol": "SGD",
        "symbol-alt-narrow": "$"
    },
    "SHP": {
        "displayName": "St. Helena-Pfund",
        "displayName-count-one": "St. Helena-Pfund",
        "displayName-count-other": "St. Helena-Pfund",
        "symbol": "SHP",
        "symbol-alt-narrow": "£"
    },
    "SIT": {
        "displayName": "Slowenischer Tolar",
        "displayName-count-one": "Slowenischer Tolar",
        "displayName-count-other": "Slowenische Tolar",
        "symbol": "SIT"
    },
    "SKK": {
        "displayName": "Slowakische Krone",
        "displayName-count-one": "Slowakische Kronen",
        "displayName-count-other": "Slowakische Kronen",
        "symbol": "SKK"
    },
    "SLL": {
        "displayName": "Sierra-leonischer Leone",
        "displayName-count-one": "Sierra-leonischer Leone",
        "displayName-count-other": "Sierra-leonische Leones",
        "symbol": "SLL"
    },
    "SOS": {
        "displayName": "Somalia-Schilling",
        "displayName-count-one": "Somalia-Schilling",
        "displayName-count-other": "Somalia-Schilling",
        "symbol": "SOS"
    },
    "SRD": {
        "displayName": "Suriname-Dollar",
        "displayName-count-one": "Suriname-Dollar",
        "displayName-count-other": "Suriname-Dollar",
        "symbol": "SRD",
        "symbol-alt-narrow": "$"
    },
    "SRG": {
        "displayName": "Suriname Gulden",
        "displayName-count-one": "Suriname-Gulden",
        "displayName-count-other": "Suriname-Gulden",
        "symbol": "SRG"
    },
    "SSP": {
        "displayName": "Südsudanesisches Pfund",
        "displayName-count-one": "Südsudanesisches Pfund",
        "displayName-count-other": "Südsudanesische Pfund",
        "symbol": "SSP",
        "symbol-alt-narrow": "£"
    },
    "STD": {

```

```

        "displayName": "São-toméischer Dobra",
        "displayName-count-one": "São-toméischer Dobra",
        "displayName-count-other": "São-toméische Dobra",
        "symbol": "STD",
        "symbol-alt-narrow": "Db"
    },
    "SUR": {
        "displayName": "Sowjetischer Rubel",
        "displayName-count-one": "Sowjetische Rubel",
        "displayName-count-other": "Sowjetische Rubel",
        "symbol": "SUR"
    },
    "SVC": {
        "displayName": "El Salvador Colon",
        "displayName-count-one": "El Salvador-Colon",
        "displayName-count-other": "El Salvador-Colon",
        "symbol": "SVC"
    },
    "SYP": {
        "displayName": "Syrisches Pfund",
        "displayName-count-one": "Syrisches Pfund",
        "displayName-count-other": "Syrische Pfund",
        "symbol": "SYP",
        "symbol-alt-narrow": "SYP"
    },
    "SZL": {
        "displayName": "Swasiländischer Lilangeni",
        "displayName-count-one": "Swasiländischer Lilangeni",
        "displayName-count-other": "Swasiländische Emalangeni",
        "symbol": "SZL"
    },
    "THB": {
        "displayName": "Thailändischer Baht",
        "displayName-count-one": "Thailändischer Baht",
        "displayName-count-other": "Thailändische Baht",
        "symbol": "฿",
        "symbol-alt-narrow": "฿"
    },
    "TJR": {
        "displayName": "Tadschikistan Rubel",
        "displayName-count-one": "Tadschikistan-Rubel",
        "displayName-count-other": "Tadschikistan-Rubel",
        "symbol": "TJR"
    },
    "TJS": {
        "displayName": "Tadschikistan-Somoni",
        "displayName-count-one": "Tadschikistan-Somoni",
        "displayName-count-other": "Tadschikistan-Somoni",
        "symbol": "TJS"
    },
    "TMM": {
        "displayName": "Turkmenistan-Manat (1993-2009)",
        "displayName-count-one": "Turkmenistan-Manat (1993-2009)",
        "displayName-count-other": "Turkmenistan-Manat (1993-2009)",
        "symbol": "TMM"
    },
    "TMT": {

```

```

        "displayName": "Turkmenistan-Manat",
        "displayName-count-one": "Turkmenistan-Manat",
        "displayName-count-other": "Turkmenistan-Manat",
        "symbol": "TMT"
    },
    "TND": {
        "displayName": "Tunesischer Dinar",
        "displayName-count-one": "Tunesischer Dinar",
        "displayName-count-other": "Tunesische Dinar",
        "symbol": "TND"
    },
    "TOP": {
        "displayName": "Tongaischer Pa'anga",
        "displayName-count-one": "Tongaischer Pa'anga",
        "displayName-count-other": "Tongaische Pa'anga",
        "symbol": "TOP",
        "symbol-alt-narrow": "T$"
    },
    "TPE": {
        "displayName": "Timor-Escudo",
        "displayName-count-one": "Timor-Escudo",
        "displayName-count-other": "Timor-Escudo",
        "symbol": "TPE"
    },
    "TRL": {
        "displayName": "Türkische Lira (1922-2005)",
        "displayName-count-one": "Türkische Lira (1922-2005)",
        "displayName-count-other": "Türkische Lira (1922-2005)",
        "symbol": "TRL"
    },
    "TRY": {
        "displayName": "Türkische Lira",
        "displayName-count-one": "Türkische Lira",
        "displayName-count-other": "Türkische Lira",
        "symbol": "TRY",
        "symbol-alt-narrow": "₺",
        "symbol-alt-variant": "TL"
    },
    "TTD": {
        "displayName": "Trinidad und Tobago-Dollar",
        "displayName-count-one": "Trinidad und Tobago-Dollar",
        "displayName-count-other": "Trinidad und Tobago-Dollar",
        "symbol": "TTD",
        "symbol-alt-narrow": "$"
    },
    "TWD": {
        "displayName": "Neuer Taiwan-Dollar",
        "displayName-count-one": "Neuer Taiwan-Dollar",
        "displayName-count-other": "Neue Taiwan-Dollar",
        "symbol": "NT$",
        "symbol-alt-narrow": "NT$"
    },
    "TZS": {
        "displayName": "Tansania-Schilling",
        "displayName-count-one": "Tansania-Schilling",
        "displayName-count-other": "Tansania-Schilling",
        "symbol": "TZS"
    }

```

```

    },
    "UAH": {
      "displayName": "Ukrainische Hrywnja",
      "displayName-count-one": "Ukrainische Hrywnja",
      "displayName-count-other": "Ukrainische Hrywen",
      "symbol": "UAH",
      "symbol-alt-narrow": "₴"
    },
    "UAK": {
      "displayName": "Ukrainischer Karbovanetz",
      "displayName-count-one": "Ukrainische Karbovanetz",
      "displayName-count-other": "Ukrainische Karbovanetz",
      "symbol": "UAK"
    },
    "UGS": {
      "displayName": "Uganda-Schilling (1966-1987)",
      "displayName-count-one": "Uganda-Schilling (1966-1987)",
      "displayName-count-other": "Uganda-Schilling (1966-1987)",
      "symbol": "UGS"
    },
    "UGX": {
      "displayName": "Uganda-Schilling",
      "displayName-count-one": "Uganda-Schilling",
      "displayName-count-other": "Uganda-Schilling",
      "symbol": "UGX"
    },
    "USD": {
      "displayName": "US-Dollar",
      "displayName-count-one": "US-Dollar",
      "displayName-count-other": "US-Dollar",
      "symbol": "$",
      "symbol-alt-narrow": "$"
    },
    "USN": {
      "displayName": "US Dollar (Nächster Tag)",
      "displayName-count-one": "US-Dollar (Nächster Tag)",
      "displayName-count-other": "US-Dollar (Nächster Tag)",
      "symbol": "USN"
    },
    "USS": {
      "displayName": "US Dollar (Gleicher Tag)",
      "displayName-count-one": "US-Dollar (Gleicher Tag)",
      "displayName-count-other": "US-Dollar (Gleicher Tag)",
      "symbol": "USS"
    },
    "UYI": {
      "displayName": "Uruguayischer Peso (Indexierte  
Rechnungseinheiten)",
      "displayName-count-one": "Uruguayischer Peso (Indexierte  
Rechnungseinheiten)",
      "displayName-count-other": "Uruguayische Pesos (Indexierte  
Rechnungseinheiten)",
      "symbol": "UYI"
    },
    "UYP": {
      "displayName": "Uruguayischer Peso (1975-1993)",
      "displayName-count-one": "Uruguayischer Peso (1975-1993)",

```

```

        "displayName-count-other": "Uruguayische Pesos (1975-1993)",
        "symbol": "UYP"
    },
    "UYU": {
        "displayName": "Uruguayischer Peso",
        "displayName-count-one": "Uruguayischer Peso",
        "displayName-count-other": "Uruguayische Pesos",
        "symbol": "UYU",
        "symbol-alt-narrow": "$"
    },
    "UZS": {
        "displayName": "Usbekistan-Sum",
        "displayName-count-one": "Usbekistan-Sum",
        "displayName-count-other": "Usbekistan-Sum",
        "symbol": "UZS"
    },
    "VEB": {
        "displayName": "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-one": "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-other": "Venezolanische Bolíva
2008)",
        "symbol": "VEB"
    },
    "VEF": {
        "displayName": "Venezolanischer Bolívar",
        "displayName-count-one": "Venezolanischer Bolívar",
        "displayName-count-other": "Venezolanische Bolíva
res",
        "symbol": "VEF",
        "symbol-alt-narrow": "Bs"
    },
    "VND": {
        "displayName": "Vietnamesischer Dong",
        "displayName-count-one": "Vietnamesischer Dong",
        "displayName-count-other": "Vietnamesische Dong",
        "symbol": "₫",
        "symbol-alt-narrow": "₫"
    },
    "VNN": {
        "displayName": "Vietnamesischer Dong(1978-1985)",
        "displayName-count-one": "Vietnamesischer Dong(1978-1985)",
        "displayName-count-other": "Vietnamesische Dong(1978-1985)",
        "symbol": "VNN"
    },
    "VUV": {
        "displayName": "Vanuatu-Vatu",
        "displayName-count-one": "Vanuatu-Vatu",
        "displayName-count-other": "Vanuatu-Vatu",
        "symbol": "VUV"
    },
    "WST": {
        "displayName": "Samoanischer Tala",
        "displayName-count-one": "Samoanischer Tala",
        "displayName-count-other": "Samoanische Tala",
        "symbol": "WST"
    },
    "XAF": {
        "displayName": "CFA-Franc (BEAC)",

```

```

        "displayName-count-one": "CFA-Franc (BEAC)",
        "displayName-count-other": "CFA-Franc (BEAC)",
        "symbol": "FCFA"
    },
    "XAG": {
        "displayName": "Unze Silber",
        "displayName-count-one": "Unze Silber",
        "displayName-count-other": "Unzen Silber",
        "symbol": "XAG"
    },
    "XAU": {
        "displayName": "Unze Gold",
        "displayName-count-one": "Unze Gold",
        "displayName-count-other": "Unzen Gold",
        "symbol": "XAU"
    },
    "XBA": {
        "displayName": "Europäische Rechnungseinheit",
        "displayName-count-one": "Europäische Rechnungseinheiten",
        "displayName-count-other": "Europäische Rechnungseinheiten",
        "symbol": "XBA"
    },
    "XBB": {
        "displayName": "Europäische Währungseinheit (XBB)",
        "displayName-count-one": "Europäische Währungseinheiten (XBB)",
        "displayName-count-other": "Europäische Währungseinheiten (XBB)",
        "symbol": "XBB"
    },
    "XBC": {
        "displayName": "Europäische Rechnungseinheit (XBC)",
        "displayName-count-one": "Europäische Rechnungseinheiten (XBC)",
        "displayName-count-other": "Europäische Rechnungseinheiten
(XBC) ",
        "symbol": "XBC"
    },
    "XBD": {
        "displayName": "Europäische Rechnungseinheit (XBD)",
        "displayName-count-one": "Europäische Rechnungseinheiten (XBD)",
        "displayName-count-other": "Europäische Rechnungseinheiten
(XBD) ",
        "symbol": "XBD"
    },
    "XCD": {
        "displayName": "Ostkaribischer Dollar",
        "displayName-count-one": "Ostkaribischer Dollar",
        "displayName-count-other": "Ostkaribische Dollar",
        "symbol": "EC$",
        "symbol-alt-narrow": "$"
    },
    "XDR": {
        "displayName": "Sonderziehungsrechte",
        "displayName-count-one": "Sonderziehungsrechte",
        "displayName-count-other": "Sonderziehungsrechte",
        "symbol": "XDR"
    },
    "XEU": {
        "displayName": "Europäische Währungseinheit (XEU)",

```



```

        "displayName-count-one": "Europäische Währungseinheiten (XEU)",
        "displayName-count-other": "Europäische Währungseinheiten (XEU)",
        "symbol": "XEU"
    },
    "XFO": {
        "displayName": "Französischer Gold-Franc",
        "displayName-count-one": "Französische Gold-Franc",
        "displayName-count-other": "Französische Gold-Franc",
        "symbol": "XFO"
    },
    "XFU": {
        "displayName": "Französischer UIC-Franc",
        "displayName-count-one": "Französische UIC-Franc",
        "displayName-count-other": "Französische UIC-Franc",
        "symbol": "XFU"
    },
    "XOF": {
        "displayName": "CFA-Franc (BCEAO)",
        "displayName-count-one": "CFA-Franc (BCEAO)",
        "displayName-count-other": "CFA-Francs (BCEAO)",
        "symbol": "CFA"
    },
    "XPD": {
        "displayName": "Unze Palladium",
        "displayName-count-one": "Unze Palladium",
        "displayName-count-other": "Unzen Palladium",
        "symbol": "XPD"
    },
    "XPF": {
        "displayName": "CFP-Franc",
        "displayName-count-one": "CFP-Franc",
        "displayName-count-other": "CFP-Franc",
        "symbol": "CFPF"
    },
    "XPT": {
        "displayName": "Unze Platin",
        "displayName-count-one": "Unze Platin",
        "displayName-count-other": "Unzen Platin",
        "symbol": "XPT"
    },
    "XRE": {
        "displayName": "RINET Funds",
        "displayName-count-one": "RINET Funds",
        "displayName-count-other": "RINET Funds",
        "symbol": "XRE"
    },
    "XSU": {
        "displayName": "SUCRE",
        "displayName-count-one": "SUCRE",
        "displayName-count-other": "SUCRE",
        "symbol": "XSU"
    },
    "XTS": {
        "displayName": "Testwährung",
        "displayName-count-one": "Testwährung",
        "displayName-count-other": "Testwährung",
        "symbol": "XTS"
    }

```

```

    },
    "XUA": {
      "displayName": "Rechnungseinheit der AfEB",
      "displayName-count-one": "Rechnungseinheit der AfEB",
      "displayName-count-other": "Rechnungseinheiten der AfEB",
      "symbol": "XUA"
    },
    "XXX": {
      "displayName": "Unbekannte Währung",
      "displayName-count-one": "(unbekannte Währung)",
      "displayName-count-other": "(unbekannte Währung)",
      "symbol": "XXX"
    },
    "YDD": {
      "displayName": "Jemen-Dinar",
      "displayName-count-one": "Jemen-Dinar",
      "displayName-count-other": "Jemen-Dinar",
      "symbol": "YDD"
    },
    "YER": {
      "displayName": "Jemen-Rial",
      "displayName-count-one": "Jemen-Rial",
      "displayName-count-other": "Jemen-Rial",
      "symbol": "YER"
    },
    "YUD": {
      "displayName": "Jugoslawischer Dinar (1966-1990)",
      "displayName-count-one": "Jugoslawischer Dinar (1966-1990)",
      "displayName-count-other": "Jugoslawische Dinar (1966-1990)",
      "symbol": "YUD"
    },
    "YUM": {
      "displayName": "Jugoslawischer Neuer Dinar (1994-2002)",
      "displayName-count-one": "Jugoslawischer Neuer Dinar (1994-2002)",
      "displayName-count-other": "Jugoslawische Neue Dinar (1994-2002)",
      "symbol": "YUM"
    },
    "YUN": {
      "displayName": "Jugoslawischer Dinar (konvertibel)",
      "displayName-count-one": "Jugoslawische Dinar (konvertibel)",
      "displayName-count-other": "Jugoslawische Dinar (konvertibel)",
      "symbol": "YUN"
    },
    "YUR": {
      "displayName": "Jugoslawischer reformierter Dinar (1992-1993)",
      "displayName-count-one": "Jugoslawischer reformierter Dinar (1992-1993)",
      "displayName-count-other": "Jugoslawische reformierte Dinar (1992-1993)",
      "symbol": "YUR"
    },
    "ZAL": {
      "displayName": "Südafrikanischer Rand (Finanz)",
      "displayName-count-one": "Südafrikanischer Rand (Finanz)",
      "displayName-count-other": "Südafrikanischer Rand (Finanz)",

```

```

    "symbol": "ZAL"
  },
  "ZAR": {
    "displayName": "Südafrikanischer Rand",
    "displayName-count-one": "Südafrikanischer Rand",
    "displayName-count-other": "Südafrikanische Rand",
    "symbol": "ZAR",
    "symbol-alt-narrow": "R"
  },
  "ZMK": {
    "displayName": "Kwacha (1968-2012)",
    "displayName-count-one": "Kwacha (1968-2012)",
    "displayName-count-other": "Kwacha (1968-2012)",
    "symbol": "ZMK"
  },
  "ZMW": {
    "displayName": "Kwacha",
    "displayName-count-one": "Kwacha",
    "displayName-count-other": "Kwacha",
    "symbol": "ZMW",
    "symbol-alt-narrow": "K"
  },
  "ZRN": {
    "displayName": "Zaire-Neuer Zaïre (1993-1998)",
    "displayName-count-one": "Zaire-Neuer Zaïre (1993-1998)",
    "displayName-count-other": "Zaire-Neue Zaïre (1993-1998)",
    "symbol": "ZRN"
  },
  "ZRZ": {
    "displayName": "Zaire-Zaïre (1971-1993)",
    "displayName-count-one": "Zaire-Zaïre (1971-1993)",
    "displayName-count-other": "Zaire-Zaïre (1971-1993)",
    "symbol": "ZRZ"
  },
  "ZWD": {
    "displayName": "Simbabwe-Dollar (1980-2008)",
    "displayName-count-one": "Simbabwe-Dollar (1980-2008)",
    "displayName-count-other": "Simbabwe-Dollar (1980-2008)",
    "symbol": "ZWD"
  },
  "ZWL": {
    "displayName": "Simbabwe-Dollar (2009)",
    "displayName-count-one": "Simbabwe-Dollar (2009)",
    "displayName-count-other": "Simbabwe-Dollar (2009)",
    "symbol": "ZWL"
  },
  "ZWR": {
    "displayName": "Simbabwe-Dollar (2008)",
    "displayName-count-one": "Simbabwe-Dollar (2008)",
    "displayName-count-other": "Simbabwe-Dollar (2008)",
    "symbol": "ZWR"
  }
}
}
}
}
}

```

CURRENCIES.JSX

```
{
  "main";
  {
    "de";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 13259 $",
          "_cldrVersion";
          "31";
        }
        "language";
        "de";
      }
      "numbers";
      {
        "currencies";
        {
          "ADP";
          {
            "displayName";
            "Andorranische Pesete",
            "displayName-count-one";
            "Andorranische Pesete",
            "displayName-count-other";
            "Andorranische Peseten",
            "symbol";
            "ADP";
          }
          "AED";
          {
            "displayName";
            "VAE-Dirham",
            "displayName-count-one";
            "VAE-Dirham",
            "displayName-count-other";
            "VAE-Dirham",
            "symbol";
            "AED";
          }
          "AFA";
          {
            "displayName";
            "Afghanische Afghani (1927-2002)",
            "displayName-count-one";
            "Afghanische Afghani (1927-2002)",
            "displayName-count-other";
            "Afghanische Afghani (1927-2002)",
            "symbol";
            "AFA";
          }
        }
      }
    }
  }
}
```

```

    }
    "AFN";
    {
        "displayName";
        "Afghanischer Afghani",
        "displayName-count-one";
        "Afghanischer Afghani",
        "displayName-count-other";
        "Afghanische Afghani",
        "symbol";
        "AFN";
    }
    "ALK";
    {
        "displayName";
        "Albanischer Lek (1946-1965)",
        "displayName-count-one";
        "Albanischer Lek (1946-1965)",
        "displayName-count-other";
        "Albanische Lek (1946-1965)";
    }
    "ALL";
    {
        "displayName";
        "Albanischer Lek",
        "displayName-count-one";
        "Albanischer Lek",
        "displayName-count-other";
        "Albanische Lek",
        "symbol";
        "ALL";
    }
    "AMD";
    {
        "displayName";
        "Armenischer Dram",
        "displayName-count-one";
        "Armenischer Dram",
        "displayName-count-other";
        "Armenische Dram",
        "symbol";
        "AMD";
    }
    "ANG";
    {
        "displayName";
        "Niederländische-Antillen-Gulden",
        "displayName-count-one";
        "Niederländische-Antillen-Gulden",
        "displayName-count-other";
        "Niederländische-Antillen-Gulden",
        "symbol";
        "ANG";
    }
    "AOA";
    {
        "displayName";

```

```

        "Angolanischer Kwanza",
        "displayName-count-one";
        "Angolanischer Kwanza",
        "displayName-count-other";
        "Angolanische Kwanza",
        "symbol";
        "AOA",
        "symbol-alt-narrow";
        "Kz";
    }
    "AOK";
    {
        "displayName";
        "Angolanischer Kwanza (1977-1990)",
        "displayName-count-one";
        "Angolanischer Kwanza (1977-1990)",
        "displayName-count-other";
        "Angolanische Kwanza (1977-1990)",
        "symbol";
        "AOK";
    }
    "AON";
    {
        "displayName";
        "Angolanischer Neuer Kwanza (1990-2000)",
        "displayName-count-one";
        "Angolanischer Neuer Kwanza (1990-2000)",
        "displayName-count-other";
        "Angolanische Neue Kwanza (1990-2000)",
        "symbol";
        "AON";
    }
    "AOR";
    {
        "displayName";
        "Angolanischer Kwanza Reajustado (1995-1999)",
        "displayName-count-one";
        "Angolanischer Kwanza Reajustado (1995-1999)",
        "displayName-count-other";
        "Angolanische Kwanza Reajustado (1995-1999)",
        "symbol";
        "AOR";
    }
    "ARA";
    {
        "displayName";
        "Argentinischer Austral",
        "displayName-count-one";
        "Argentinischer Austral",
        "displayName-count-other";
        "Argentinische Austral",
        "symbol";
        "ARA";
    }
    "ARL";
    {
        "displayName";

```

```

        "Argentinischer Peso Ley (1970-1983)",
        "displayName-count-one";
        "Argentinischer Peso Ley (1970-1983)",
        "displayName-count-other";
        "Argentinische Pesos Ley (1970-1983)",
        "symbol";
        "ARL";
    }
    "ARM";
    {
        "displayName";
        "Argentinischer Peso (1881-1970)",
        "displayName-count-one";
        "Argentinischer Peso (1881-1970)",
        "displayName-count-other";
        "Argentinische Pesos (1881-1970)",
        "symbol";
        "ARM";
    }
    "ARP";
    {
        "displayName";
        "Argentinischer Peso (1983-1985)",
        "displayName-count-one";
        "Argentinischer Peso (1983-1985)",
        "displayName-count-other";
        "Argentinische Pesos (1983-1985)",
        "symbol";
        "ARP";
    }
    "ARS";
    {
        "displayName";
        "Argentinischer Peso",
        "displayName-count-one";
        "Argentinischer Peso",
        "displayName-count-other";
        "Argentinische Pesos",
        "symbol";
        "ARS",
        "symbol-alt-narrow";
        "$";
    }
    "ATS";
    {
        "displayName";
        "Österreichischer Schilling",
        "displayName-count-one";
        "Österreichischer Schilling",
        "displayName-count-other";
        "Österreichische Schilling",
        "symbol";
        "ÖS";
    }
    "AUD";
    {
        "displayName";

```

```

        "Australischer Dollar",
        "displayName-count-one";
    "Australischer Dollar",
        "displayName-count-other";
    "Australische Dollar",
        "symbol";
    "AU$",
        "symbol-alt-narrow";
    "$";
}
"AWG";
{
    "displayName";
    "Aruba-Florin",
        "displayName-count-one";
    "Aruba-Florin",
        "displayName-count-other";
    "Aruba-Florin",
        "symbol";
    "AWG";
}
"AZM";
{
    "displayName";
    "Aserbaidtschan-Manat (1993-2006)",
        "displayName-count-one";
    "Aserbaidtschan-Manat (1993-2006)",
        "displayName-count-other";
    "Aserbaidtschan-Manat (1993-2006)",
        "symbol";
    "AZM";
}
"AZN";
{
    "displayName";
    "Aserbaidtschan-Manat",
        "displayName-count-one";
    "Aserbaidtschan-Manat",
        "displayName-count-other";
    "Aserbaidtschan-Manat",
        "symbol";
    "AZN";
}
"BAD";
{
    "displayName";
    "Bosnien und Herzegowina Dinar (1992-1994)",
        "displayName-count-one";
    "Bosnien und Herzegowina Dinar (1992-1994)",
        "displayName-count-other";
    "Bosnien und Herzegowina Dinar (1992-1994)",
        "symbol";
    "BAD";
}
"BAM";
{
    "displayName";

```



```

        "Bosnien und Herzegowina Konvertierbare Mark",
        "displayName-count-one";
        "Bosnien und Herzegowina Konvertierbare Mark",
        "displayName-count-other";
        "Bosnien und Herzegowina Konvertierbare Mark",
        "symbol";
        "BAM",
        "symbol-alt-narrow";
        "KM";
    }
    "BAN";
    {
        "displayName";
        "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
        "displayName-count-one";
        "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
        "displayName-count-other";
        "Bosnien und Herzegowina Neue Dinar (1994-1997)",
        "symbol";
        "BAN";
    }
    "BBD";
    {
        "displayName";
        "Barbados-Dollar",
        "displayName-count-one";
        "Barbados-Dollar",
        "displayName-count-other";
        "Barbados-Dollar",
        "symbol";
        "BBD",
        "symbol-alt-narrow";
        "$";
    }
    "BDT";
    {
        "displayName";
        "Bangladesch-Taka",
        "displayName-count-one";
        "Bangladesch-Taka",
        "displayName-count-other";
        "Bangladesch-Taka",
        "symbol";
        "BDT",
        "symbol-alt-narrow";
        "৳";
    }
    "BEC";
    {
        "displayName";
        "Belgischer Franc (konvertibel)",
        "displayName-count-one";
        "Belgischer Franc (konvertibel)",
        "displayName-count-other";
        "Belgische Franc (konvertibel)",
        "symbol";
    }

```

```

        "BEC";
    }
    "BEF";
    {
        "displayName";
        "Belgischer Franc",
        "displayName-count-one";
        "Belgischer Franc",
        "displayName-count-other";
        "Belgische Franc",
        "symbol";
        "BEF";
    }
    "BEL";
    {
        "displayName";
        "Belgischer Finanz-Franc",
        "displayName-count-one";
        "Belgischer Finanz-Franc",
        "displayName-count-other";
        "Belgische Finanz-Franc",
        "symbol";
        "BEL";
    }
    "BGL";
    {
        "displayName";
        "Bulgarische Lew (1962-1999)",
        "displayName-count-one";
        "Bulgarische Lew (1962-1999)",
        "displayName-count-other";
        "Bulgarische Lew (1962-1999)",
        "symbol";
        "BGL";
    }
    "BGM";
    {
        "displayName";
        "Bulgarischer Lew (1952-1962)",
        "displayName-count-one";
        "Bulgarischer Lew (1952-1962)",
        "displayName-count-other";
        "Bulgarische Lew (1952-1962)",
        "symbol";
        "BGK";
    }
    "BGN";
    {
        "displayName";
        "Bulgarischer Lew",
        "displayName-count-one";
        "Bulgarischer Lew",
        "displayName-count-other";
        "Bulgarische Lew",
        "symbol";
        "BGN";
    }
}

```

```

    "BGO";
    {
      "displayName";
      "Bulgarischer Lew (1879-1952)",
      "displayName-count-one";
      "Bulgarischer Lew (1879-1952)",
      "displayName-count-other";
      "Bulgarische Lew (1879-1952)",
      "symbol";
      "BGJ";
    }
    "BHD";
    {
      "displayName";
      "Bahrain-Dinar",
      "displayName-count-one";
      "Bahrain-Dinar",
      "displayName-count-other";
      "Bahrain-Dinar",
      "symbol";
      "BHD";
    }
    "BIF";
    {
      "displayName";
      "Burundi-Franc",
      "displayName-count-one";
      "Burundi-Franc",
      "displayName-count-other";
      "Burundi-Francis",
      "symbol";
      "BIF";
    }
    "BMD";
    {
      "displayName";
      "Bermuda-Dollar",
      "displayName-count-one";
      "Bermuda-Dollar",
      "displayName-count-other";
      "Bermuda-Dollar",
      "symbol";
      "BMD",
      "symbol-alt-narrow";
      "$";
    }
    "BND";
    {
      "displayName";
      "Brunei-Dollar",
      "displayName-count-one";
      "Brunei-Dollar",
      "displayName-count-other";
      "Brunei-Dollar",
      "symbol";
      "BND",
      "symbol-alt-narrow";
    }

```

```

        "$";
    }
    "BOB";
    {
        "displayName";
        "Bolivanischer Boliviano",
        "displayName-count-one";
        "Bolivanischer Boliviano",
        "displayName-count-other";
        "Bolivianische Bolivianos",
        "symbol";
        "BOB",
        "symbol-alt-narrow";
        "Bs";
    }
    "BOL";
    {
        "displayName";
        "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-one";
        "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-other";
        "Bolivianische Bolivianos (1863-1963)",
        "symbol";
        "BOL";
    }
    "BOP";
    {
        "displayName";
        "Bolivianischer Peso",
        "displayName-count-one";
        "Bolivianischer Peso",
        "displayName-count-other";
        "Bolivianische Peso",
        "symbol";
        "BOP";
    }
    "BOV";
    {
        "displayName";
        "Boliviansiche Mvdol",
        "displayName-count-one";
        "Boliviansiche Mvdol",
        "displayName-count-other";
        "Bolivianische Mvdol",
        "symbol";
        "BOV";
    }
    "BRB";
    {
        "displayName";
        "Brasilianischer Cruzeiro Novo (1967-1986)",
        "displayName-count-one";
        "Brasilianischer Cruzeiro Novo (1967-1986)",
        "displayName-count-other";
        "Brasilianische Cruzeiro Novo (1967-1986)",
        "symbol";
    }

```

```

        "BRB";
    }
    "BRC";
    {
        "displayName";
        "Brasilianischer Cruzado (1986-1989)",
        "displayName-count-one";
        "Brasilianischer Cruzado (1986-1989)",
        "displayName-count-other";
        "Brasilianische Cruzado (1986-1989)",
        "symbol";
        "BRC";
    }
    "BRE";
    {
        "displayName";
        "Brasilianischer Cruzeiro (1990-1993)",
        "displayName-count-one";
        "Brasilianischer Cruzeiro (1990-1993)",
        "displayName-count-other";
        "Brasilianische Cruzeiro (1990-1993)",
        "symbol";
        "BRE";
    }
    "BRL";
    {
        "displayName";
        "Brasilianischer Real",
        "displayName-count-one";
        "Brasilianischer Real",
        "displayName-count-other";
        "Brasilianische Real",
        "symbol";
        "R$",
        "symbol-alt-narrow";
        "R$";
    }
    "BRN";
    {
        "displayName";
        "Brasilianischer Cruzado Novo (1989-1990)",
        "displayName-count-one";
        "Brasilianischer Cruzado Novo (1989-1990)",
        "displayName-count-other";
        "Brasilianische Cruzado Novo (1989-1990)",
        "symbol";
        "BRN";
    }
    "BRR";
    {
        "displayName";
        "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-one";
        "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-other";
        "Brasilianische Cruzeiro (1993-1994)",
        "symbol";
    }

```

```

        "BRR";
    }
    "BRZ";
    {
        "displayName";
        "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-one";
        "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-other";
        "Brasilianischer Cruzeiro (1942-1967)",
        "symbol";
        "BRZ";
    }
    "BSD";
    {
        "displayName";
        "Bahamas-Dollar",
        "displayName-count-one";
        "Bahamas-Dollar",
        "displayName-count-other";
        "Bahamas-Dollar",
        "symbol";
        "BSD",
        "symbol-alt-narrow";
        "$";
    }
    "BTN";
    {
        "displayName";
        "Bhutan-Ngultrum",
        "displayName-count-one";
        "Bhutan-Ngultrum",
        "displayName-count-other";
        "Bhutan-Ngultrum",
        "symbol";
        "BTN";
    }
    "BUK";
    {
        "displayName";
        "Birmanischer Kyat",
        "displayName-count-one";
        "Birmanischer Kyat",
        "displayName-count-other";
        "Birmanische Kyat",
        "symbol";
        "BUK";
    }
    "BWP";
    {
        "displayName";
        "Botswanischer Pula",
        "displayName-count-one";
        "Botswanischer Pula",
        "displayName-count-other";
        "Botswanische Pula",
        "symbol";
    }

```

```

        "BWP",
        "symbol-alt-narrow";
        "P";
    }
    "BYB";
    {
        "displayName";
        "Belarus-Rubel (1994-1999)",
        "displayName-count-one";
        "Belarus-Rubel (1994-1999)",
        "displayName-count-other";
        "Belarus-Rubel (1994-1999)",
        "symbol";
        "BYB";
    }
    "BYN";
    {
        "displayName";
        "Weißrussischer Rubel",
        "displayName-count-one";
        "Weißrussischer Rubel",
        "displayName-count-other";
        "Weißrussische Rubel",
        "symbol";
        "BYN",
        "symbol-alt-narrow";
        "p.";
    }
    "BYR";
    {
        "displayName";
        "Weißrussischer Rubel (2000-2016)",
        "displayName-count-one";
        "Weißrussischer Rubel (2000-2016)",
        "displayName-count-other";
        "Weißrussische Rubel (2000-2016)",
        "symbol";
        "BYR";
    }
    "BZD";
    {
        "displayName";
        "Belize-Dollar",
        "displayName-count-one";
        "Belize-Dollar",
        "displayName-count-other";
        "Belize-Dollar",
        "symbol";
        "BZD",
        "symbol-alt-narrow";
        "$";
    }
    "CAD";
    {
        "displayName";
        "Kanadischer Dollar",
        "displayName-count-one";

```

```

        "Kanadischer Dollar",
        "displayName-count-other";
    "Kanadische Dollar",
        "symbol";
    "CA$",
        "symbol-alt-narrow";
    "$";
}
"CDF";
{
    "displayName";
    "Kongo-Franc",
        "displayName-count-one";
    "Kongo-Franc",
        "displayName-count-other";
    "Kongo-Francs",
        "symbol";
    "CDF";
}
"CHE";
{
    "displayName";
    "WIR-Euro",
        "displayName-count-one";
    "WIR-Euro",
        "displayName-count-other";
    "WIR-Euro",
        "symbol";
    "CHE";
}
"CHF";
{
    "displayName";
    "Schweizer Franken",
        "displayName-count-one";
    "Schweizer Franken",
        "displayName-count-other";
    "Schweizer Franken",
        "symbol";
    "CHF";
}
"CHW";
{
    "displayName";
    "WIR Franken",
        "displayName-count-one";
    "WIR Franken",
        "displayName-count-other";
    "WIR Franken",
        "symbol";
    "CHW";
}
"CLE";
{
    "displayName";
    "Chilenischer Escudo",
        "displayName-count-one";

```



```

        "Chilenischer Escudo",
        "displayName-count-other";
        "Chilenische Escudo",
        "symbol";
        "CLE";
    }
    "CLF";
    {
        "displayName";
        "Chilenische Unidades de Fomento",
        "displayName-count-one";
        "Chilenische Unidades de Fomento",
        "displayName-count-other";
        "Chilenische Unidades de Fomento",
        "symbol";
        "CLF";
    }
    "CLP";
    {
        "displayName";
        "Chilenischer Peso",
        "displayName-count-one";
        "Chilenischer Peso",
        "displayName-count-other";
        "Chilenische Pesos",
        "symbol";
        "CLP",
        "symbol-alt-narrow";
        "$";
    }
    "CNX";
    {
        "displayName";
        "Dollar der Chinesischen Volksbank",
        "displayName-count-one";
        "Dollar der Chinesischen Volksbank",
        "displayName-count-other";
        "Dollar der Chinesischen Volksbank",
        "symbol";
        "CNX";
    }
    "CNY";
    {
        "displayName";
        "Renminbi Yuan",
        "displayName-count-one";
        "Chinesischer Yuan",
        "displayName-count-other";
        "Renminbi Yuan",
        "symbol";
        "CN¥",
        "symbol-alt-narrow";
        "¥";
    }
    "COP";
    {
        "displayName";

```

```

        "Kolumbianischer Peso",
        "displayName-count-one";
        "Kolumbianischer Peso",
        "displayName-count-other";
        "Kolumbianische Pesos",
        "symbol";
        "COP",
        "symbol-alt-narrow";
        "$";
    }
    "COU";
    {
        "displayName";
        "Kolumbianische Unidades de valor real",
        "displayName-count-one";
        "Kolumbianische Unidad de valor real",
        "displayName-count-other";
        "Kolumbianische Unidades de valor real",
        "symbol";
        "COU";
    }
    "CRC";
    {
        "displayName";
        "Costa-Rica-Colón",
        "displayName-count-one";
        "Costa-Rica-Colón",
        "displayName-count-other";
        "Costa-Rica-Colón",
        "symbol";
        "CRC",
        "symbol-alt-narrow";
        "₡";
    }
    "CSD";
    {
        "displayName";
        "Serbischer Dinar (2002-2006)",
        "displayName-count-one";
        "Serbischer Dinar (2002-2006)",
        "displayName-count-other";
        "Serbische Dinar (2002-2006)",
        "symbol";
        "CSD";
    }
    "CSK";
    {
        "displayName";
        "Tschechoslowakische Krone",
        "displayName-count-one";
        "Tschechoslowakische Kronen",
        "displayName-count-other";
        "Tschechoslowakische Kronen",
        "symbol";
        "CSK";
    }
    "CUC";

```

```

    {
      "displayName";
      "Kubanischer Peso (konvertibel)",
        "displayName-count-one";
      "Kubanischer Peso (konvertibel)",
        "displayName-count-other";
      "Kubanische Pesos (konvertibel)",
        "symbol";
      "CUC",
        "symbol-alt-narrow";
      "Cub$";
    }
    "CUP";
    {
      "displayName";
      "Kubanischer Peso",
        "displayName-count-one";
      "Kubanischer Peso",
        "displayName-count-other";
      "Kubanische Pesos",
        "symbol";
      "CUP",
        "symbol-alt-narrow";
      "$";
    }
    "CVE";
    {
      "displayName";
      "Cabo-Verde-Escudo",
        "displayName-count-one";
      "Cabo-Verde-Escudo",
        "displayName-count-other";
      "Cabo-Verde-Escudos",
        "symbol";
      "CVE";
    }
    "CYP";
    {
      "displayName";
      "Zypern-Pfund",
        "displayName-count-one";
      "Zypern Pfund",
        "displayName-count-other";
      "Zypern Pfund",
        "symbol";
      "CYP";
    }
    "CZK";
    {
      "displayName";
      "Tschechische Krone",
        "displayName-count-one";
      "Tschechische Krone",
        "displayName-count-other";
      "Tschechische Kronen",
        "symbol";
      "CZK",

```

```

        "symbol-alt-narrow";
        "Kč";
    }
    "DDM";
    {
        "displayName";
        "Mark der DDR",
        "displayName-count-one";
        "Mark der DDR",
        "displayName-count-other";
        "Mark der DDR",
        "symbol";
        "DDM";
    }
    "DEM";
    {
        "displayName";
        "Deutsche Mark",
        "displayName-count-one";
        "Deutsche Mark",
        "displayName-count-other";
        "Deutsche Mark",
        "symbol";
        "DM";
    }
    "DJF";
    {
        "displayName";
        "Dschibuti-Franc",
        "displayName-count-one";
        "Dschibuti-Franc",
        "displayName-count-other";
        "Dschibuti-Franc",
        "symbol";
        "DJF";
    }
    "DKK";
    {
        "displayName";
        "Dänische Krone",
        "displayName-count-one";
        "Dänische Krone",
        "displayName-count-other";
        "Dänische Kronen",
        "symbol";
        "DKK",
        "symbol-alt-narrow";
        "kr";
    }
    "DOP";
    {
        "displayName";
        "Dominikanischer Peso",
        "displayName-count-one";
        "Dominikanischer Peso",
        "displayName-count-other";
        "Dominikanische Pesos",

```

```

        "symbol";
        "DOP",
        "symbol-alt-narrow";
        "$";
    }
    "DZD";
    {
        "displayName";
        "Algerischer Dinar",
        "displayName-count-one";
        "Algerischer Dinar",
        "displayName-count-other";
        "Algerische Dinar",
        "symbol";
        "DZD";
    }
    "ECS";
    {
        "displayName";
        "Ecuadorianischer Sucre",
        "displayName-count-one";
        "Ecuadorianischer Sucre",
        "displayName-count-other";
        "Ecuadorianische Sucre",
        "symbol";
        "ECS";
    }
    "ECV";
    {
        "displayName";
        "Verrechnungseinheit für Ecuador",
        "displayName-count-one";
        "Verrechnungseinheiten für Ecuador",
        "displayName-count-other";
        "Verrechnungseinheiten für Ecuador",
        "symbol";
        "ECV";
    }
    "EEK";
    {
        "displayName";
        "Estnische Krone",
        "displayName-count-one";
        "Estnische Krone",
        "displayName-count-other";
        "Estnische Kronen",
        "symbol";
        "EEK";
    }
    "EGP";
    {
        "displayName";
        "Ägyptisches Pfund",
        "displayName-count-one";
        "Ägyptisches Pfund",
        "displayName-count-other";
        "Ägyptische Pfund",

```

```

        "symbol";
        "EGP",
        "symbol-alt-narrow";
        "£";
    }
    "ERN";
    {
        "displayName";
        "Eritreischer Nakfa",
        "displayName-count-one";
        "Eritreischer Nakfa",
        "displayName-count-other";
        "Eritreische Nakfa",
        "symbol";
        "ERN";
    }
    "ESA";
    {
        "displayName";
        "Spanische Peseta (A-Konten)",
        "displayName-count-one";
        "Spanische Peseta (A-Konten)",
        "displayName-count-other";
        "Spanische Peseten (A-Konten)",
        "symbol";
        "ESA";
    }
    "ESB";
    {
        "displayName";
        "Spanische Peseta (konvertibel)",
        "displayName-count-one";
        "Spanische Peseta (konvertibel)",
        "displayName-count-other";
        "Spanische Peseten (konvertibel)",
        "symbol";
        "ESB";
    }
    "ESP";
    {
        "displayName";
        "Spanische Peseta",
        "displayName-count-one";
        "Spanische Peseta",
        "displayName-count-other";
        "Spanische Peseten",
        "symbol";
        "ESP",
        "symbol-alt-narrow";
        "₧";
    }
    "ETB";
    {
        "displayName";
        "Äthiopischer Birr",
        "displayName-count-one";
        "Äthiopischer Birr",

```

```

        "displayName-count-other";
        "Äthiopische Birr",
        "symbol";
        "ETB";
    }
    "EUR";
    {
        "displayName";
        "Euro",
        "displayName-count-one";
        "Euro",
        "displayName-count-other";
        "Euro",
        "symbol";
        "€",
        "symbol-alt-narrow";
        "€";
    }
    "FIM";
    {
        "displayName";
        "Finnische Mark",
        "displayName-count-one";
        "Finnische Mark",
        "displayName-count-other";
        "Finnische Mark",
        "symbol";
        "FIM";
    }
    "FJD";
    {
        "displayName";
        "Fidschi-Dollar",
        "displayName-count-one";
        "Fidschi-Dollar",
        "displayName-count-other";
        "Fidschi-Dollar",
        "symbol";
        "FJD",
        "symbol-alt-narrow";
        "$";
    }
    "FKP";
    {
        "displayName";
        "Falkland-Pfund",
        "displayName-count-one";
        "Falkland-Pfund",
        "displayName-count-other";
        "Falkland-Pfund",
        "symbol";
        "FKP",
        "symbol-alt-narrow";
        "Fl£";
    }
    "FRF";
    {

```

```

        "displayName";
        "Französischer Franc",
        "displayName-count-one";
        "Französischer Franc",
        "displayName-count-other";
        "Französische Franc",
        "symbol";
        "FRF";
    }
    "GBP";
    {
        "displayName";
        "Britisches Pfund",
        "displayName-count-one";
        "Britisches Pfund",
        "displayName-count-other";
        "Britische Pfund",
        "symbol";
        "£",
        "symbol-alt-narrow";
        "£";
    }
    "GEK";
    {
        "displayName";
        "Georgischer Kupon Larit",
        "displayName-count-one";
        "Georgischer Kupon Larit",
        "displayName-count-other";
        "Georgische Kupon Larit",
        "symbol";
        "GEK";
    }
    "GEL";
    {
        "displayName";
        "Georgischer Lari",
        "displayName-count-one";
        "Georgischer Lari",
        "displayName-count-other";
        "Georgische Lari",
        "symbol";
        "GEL",
        "symbol-alt-narrow";
        "ლ",
        "symbol-alt-variant";
        "ლ";
    }
    "GHC";
    {
        "displayName";
        "Ghanaischer Cedi (1979-2007)",
        "displayName-count-one";
        "Ghanaischer Cedi (1979-2007)",
        "displayName-count-other";
        "Ghanaische Cedi (1979-2007)",
        "symbol";
    }

```



```

        "GHC";
    }
    "GHS";
    {
        "displayName";
        "Ghanaischer Cedi",
        "displayName-count-one";
        "Ghanaischer Cedi",
        "displayName-count-other";
        "Ghanaische Cedi",
        "symbol";
        "GHS";
    }
    "GIP";
    {
        "displayName";
        "Gibraltar-Pfund",
        "displayName-count-one";
        "Gibraltar-Pfund",
        "displayName-count-other";
        "Gibraltar Pfund",
        "symbol";
        "GIP",
        "symbol-alt-narrow";
        "£";
    }
    "GMD";
    {
        "displayName";
        "Gambia-Dalasi",
        "displayName-count-one";
        "Gambia-Dalasi",
        "displayName-count-other";
        "Gambia-Dalasi",
        "symbol";
        "GMD";
    }
    "GNF";
    {
        "displayName";
        "Guinea-Franc",
        "displayName-count-one";
        "Guinea-Franc",
        "displayName-count-other";
        "Guinea-Franc",
        "symbol";
        "GNF",
        "symbol-alt-narrow";
        "F.G.";
    }
    "GNS";
    {
        "displayName";
        "Guineischer Syli",
        "displayName-count-one";
        "Guineischer Syli",
        "displayName-count-other";
    }

```

```

        "Guineische Syli",
        "symbol";
        "GNS";
    }
    "GQE";
    {
        "displayName";
        "Äquatorialguinea-Ekwele",
        "displayName-count-one";
        "Äquatorialguinea-Ekwele",
        "displayName-count-other";
        "Äquatorialguinea-Ekwele",
        "symbol";
        "GQE";
    }
    "GRD";
    {
        "displayName";
        "Griechische Drachme",
        "displayName-count-one";
        "Griechische Drachme",
        "displayName-count-other";
        "Griechische Drachmen",
        "symbol";
        "GRD";
    }
    "GTQ";
    {
        "displayName";
        "Guatemaltekinscher Quetzal",
        "displayName-count-one";
        "Guatemaltekinscher Quetzal",
        "displayName-count-other";
        "Guatemaltekinsche Quetzales",
        "symbol";
        "GTQ",
        "symbol-alt-narrow";
        "Q";
    }
    "GWE";
    {
        "displayName";
        "Portugiesisch Guinea Escudo",
        "displayName-count-one";
        "Portugiesisch Guinea Escudo",
        "displayName-count-other";
        "Portugiesisch Guinea Escudo",
        "symbol";
        "GWE";
    }
    "GWP";
    {
        "displayName";
        "Guinea-Bissau Peso",
        "displayName-count-one";
        "Guinea-Bissau Peso",
        "displayName-count-other";
    }

```

```

        "Guinea-Bissau Pesos",
        "symbol";
        "GWP";
    }
    "GYD";
    {
        "displayName";
        "Guyana-Dollar",
        "displayName-count-one";
        "Guyana-Dollar",
        "displayName-count-other";
        "Guyana-Dollar",
        "symbol";
        "GYD",
        "symbol-alt-narrow";
        "$";
    }
    "HKD";
    {
        "displayName";
        "Hongkong-Dollar",
        "displayName-count-one";
        "Hongkong-Dollar",
        "displayName-count-other";
        "Hongkong-Dollar",
        "symbol";
        "HK$",
        "symbol-alt-narrow";
        "$";
    }
    "HNL";
    {
        "displayName";
        "Honduras-Lempira",
        "displayName-count-one";
        "Honduras-Lempira",
        "displayName-count-other";
        "Honduras-Lempira",
        "symbol";
        "HNL",
        "symbol-alt-narrow";
        "L";
    }
    "HRD";
    {
        "displayName";
        "Kroatischer Dinar",
        "displayName-count-one";
        "Kroatischer Dinar",
        "displayName-count-other";
        "Kroatische Dinar",
        "symbol";
        "HRD";
    }
    "HRK";
    {
        "displayName";

```

```

        "Kroatischer Kuna",
        "displayName-count-one";
        "Kroatischer Kuna",
        "displayName-count-other";
        "Kroatische Kuna",
        "symbol";
        "HRK",
        "symbol-alt-narrow";
        "kn";
    }
    "HTG";
    {
        "displayName";
        "Haitianische Gourde",
        "displayName-count-one";
        "Haitianische Gourde",
        "displayName-count-other";
        "Haitianische Gourdes",
        "symbol";
        "HTG";
    }
    "HUF";
    {
        "displayName";
        "Ungarischer Forint",
        "displayName-count-one";
        "Ungarischer Forint",
        "displayName-count-other";
        "Ungarische Forint",
        "symbol";
        "HUF",
        "symbol-alt-narrow";
        "Ft";
    }
    "IDR";
    {
        "displayName";
        "Indonesische Rupiah",
        "displayName-count-one";
        "Indonesische Rupiah",
        "displayName-count-other";
        "Indonesische Rupiah",
        "symbol";
        "IDR",
        "symbol-alt-narrow";
        "Rp";
    }
    "IEP";
    {
        "displayName";
        "Irisches Pfund",
        "displayName-count-one";
        "Irisches Pfund",
        "displayName-count-other";
        "Irische Pfund",
        "symbol";
        "IEP";
    }

```

```

    }
    "ILP";
    {
        "displayName";
        "Israelisches Pfund",
        "displayName-count-one";
        "Israelisches Pfund",
        "displayName-count-other";
        "Israelische Pfund",
        "symbol";
        "ILP";
    }
    "ILR";
    {
        "displayName";
        "Israelischer Schekel (1980-1985)",
        "displayName-count-one";
        "Israelischer Schekel (1980-1985)",
        "displayName-count-other";
        "Israelische Schekel (1980-1985)";
    }
    "ILS";
    {
        "displayName";
        "Israelischer Neuer Schekel",
        "displayName-count-one";
        "Israelischer Neuer Schekel",
        "displayName-count-other";
        "Israelische Neue Schekel",
        "symbol";
        "₪",
        "symbol-alt-narrow";
        "₪";
    }
    "INR";
    {
        "displayName";
        "Indische Rupie",
        "displayName-count-one";
        "Indische Rupie",
        "displayName-count-other";
        "Indische Rupien",
        "symbol";
        "₹",
        "symbol-alt-narrow";
        "₹";
    }
    "IQD";
    {
        "displayName";
        "Irakischer Dinar",
        "displayName-count-one";
        "Irakischer Dinar",
        "displayName-count-other";
        "Irakische Dinar",
        "symbol";
        "IQD";
    }

```

```
}
"IRR";
{
  "displayName";
  "Iranischer Rial",
    "displayName-count-one";
  "Iranischer Rial",
    "displayName-count-other";
  "Iranische Rial",
    "symbol";
  "IRR";
}
"ISJ";
{
  "displayName";
  "Isländische Krone (1918-1981)",
    "displayName-count-one";
  "Isländische Krone (1918-1981)",
    "displayName-count-other";
  "Isländische Kronen (1918-1981)";
}
"ISK";
{
  "displayName";
  "Isländische Krone",
    "displayName-count-one";
  "Isländische Krone",
    "displayName-count-other";
  "Isländische Kronen",
    "symbol";
  "ISK",
    "symbol-alt-narrow";
  "kr";
}
"ITL";
{
  "displayName";
  "Italienische Lira",
    "displayName-count-one";
  "Italienische Lira",
    "displayName-count-other";
  "Italienische Lire",
    "symbol";
  "ITL";
}
"JMD";
{
  "displayName";
  "Jamaika-Dollar",
    "displayName-count-one";
  "Jamaika-Dollar",
    "displayName-count-other";
  "Jamaika-Dollar",
    "symbol";
  "JMD",
    "symbol-alt-narrow";
  "$";
}
```

```

    }
    "JOD";
    {
        "displayName";
        "Jordanischer Dinar",
        "displayName-count-one";
        "Jordanischer Dinar",
        "displayName-count-other";
        "Jordanische Dinar",
        "symbol";
        "JOD";
    }
    "JPY";
    {
        "displayName";
        "Japanischer Yen",
        "displayName-count-one";
        "Japanischer Yen",
        "displayName-count-other";
        "Japanische Yen",
        "symbol";
        "¥",
        "symbol-alt-narrow";
        "¥";
    }
    "KES";
    {
        "displayName";
        "Kenia-Schilling",
        "displayName-count-one";
        "Kenia-Schilling",
        "displayName-count-other";
        "Kenia-Schilling",
        "symbol";
        "KES";
    }
    "KGS";
    {
        "displayName";
        "Kirgisischer Som",
        "displayName-count-one";
        "Kirgisischer Som",
        "displayName-count-other";
        "Kirgisische Som",
        "symbol";
        "KGS";
    }
    "KHR";
    {
        "displayName";
        "Kambodschanischer Riel",
        "displayName-count-one";
        "Kambodschanischer Riel",
        "displayName-count-other";
        "Kambodschanische Riel",
        "symbol";
        "KHR",

```

```

        "symbol-alt-narrow";
        "₭";
    }
    "KMF";
    {
        "displayName";
        "Komoren-Franc",
        "displayName-count-one";
        "Komoren-Franc",
        "displayName-count-other";
        "Komoren-Francs",
        "symbol";
        "KMF",
        "symbol-alt-narrow";
        "FC";
    }
    "KPW";
    {
        "displayName";
        "Nordkoreanischer Won",
        "displayName-count-one";
        "Nordkoreanischer Won",
        "displayName-count-other";
        "Nordkoreanische Won",
        "symbol";
        "KPW",
        "symbol-alt-narrow";
        "₩";
    }
    "KRH";
    {
        "displayName";
        "Südkoreanischer Hwan (1953-1962)",
        "displayName-count-one";
        "Südkoreanischer Hwan (1953-1962)",
        "displayName-count-other";
        "Südkoreanischer Hwan (1953-1962)",
        "symbol";
        "KRH";
    }
    "KRO";
    {
        "displayName";
        "Südkoreanischer Won (1945-1953)",
        "displayName-count-one";
        "Südkoreanischer Won (1945-1953)",
        "displayName-count-other";
        "Südkoreanischer Won (1945-1953)",
        "symbol";
        "KRO";
    }
    "KRW";
    {
        "displayName";
        "Südkoreanischer Won",
        "displayName-count-one";

```



```

        "Südkoreanischer Won",
        "displayName-count-other";
    "Südkoreanische Won",
        "symbol";
    "₩",
        "symbol-alt-narrow";
    "₩";
}
"KWD";
{
    "displayName";
    "Kuwait-Dinar",
        "displayName-count-one";
    "Kuwait-Dinar",
        "displayName-count-other";
    "Kuwait-Dinar",
        "symbol";
    "KWD";
}
"KYD";
{
    "displayName";
    "Kaiman-Dollar",
        "displayName-count-one";
    "Kaiman-Dollar",
        "displayName-count-other";
    "Kaiman-Dollar",
        "symbol";
    "KYD",
        "symbol-alt-narrow";
    "$";
}
"KZT";
{
    "displayName";
    "Kasachischer Tenge",
        "displayName-count-one";
    "Kasachischer Tenge",
        "displayName-count-other";
    "Kasachische Tenge",
        "symbol";
    "KZT",
        "symbol-alt-narrow";
    "₸";
}
"LAK";
{
    "displayName";
    "Laotischer Kip",
        "displayName-count-one";
    "Laotischer Kip",
        "displayName-count-other";
    "Laotische Kip",
        "symbol";
    "LAK",
        "symbol-alt-narrow";
    "₭";
}

```

```

    }
    "LBP";
    {
        "displayName";
        "Libanesisches Pfund",
        "displayName-count-one";
        "Libanesisches Pfund",
        "displayName-count-other";
        "Libanesische Pfund",
        "symbol";
        "LBP",
        "symbol-alt-narrow";
        "L£";
    }
    "LKR";
    {
        "displayName";
        "Sri-Lanka-Rupie",
        "displayName-count-one";
        "Sri-Lanka-Rupie",
        "displayName-count-other";
        "Sri-Lanka-Rupien",
        "symbol";
        "LKR",
        "symbol-alt-narrow";
        "Rs";
    }
    "LRD";
    {
        "displayName";
        "Liberianischer Dollar",
        "displayName-count-one";
        "Liberianischer Dollar",
        "displayName-count-other";
        "Liberianische Dollar",
        "symbol";
        "LRD",
        "symbol-alt-narrow";
        "$";
    }
    "LSL";
    {
        "displayName";
        "Loti",
        "displayName-count-one";
        "Loti",
        "displayName-count-other";
        "Loti",
        "symbol";
        "LSL";
    }
    "LTL";
    {
        "displayName";
        "Litauischer Litas",
        "displayName-count-one";
        "Litauischer Litas",

```

```

        "displayName-count-other";
        "Litauische Litas",
        "symbol";
        "LTL",
        "symbol-alt-narrow";
        "Lt";
    }
    "LTT";
    {
        "displayName";
        "Litauischer Talonas",
        "displayName-count-one";
        "Litauische Talonas",
        "displayName-count-other";
        "Litauische Talonas",
        "symbol";
        "LTT";
    }
    "LUC";
    {
        "displayName";
        "Luxemburgischer Franc (konvertibel)",
        "displayName-count-one";
        "Luxemburgische Franc (konvertibel)",
        "displayName-count-other";
        "Luxemburgische Franc (konvertibel)",
        "symbol";
        "LUC";
    }
    "LUF";
    {
        "displayName";
        "Luxemburgischer Franc",
        "displayName-count-one";
        "Luxemburgische Franc",
        "displayName-count-other";
        "Luxemburgische Franc",
        "symbol";
        "LUF";
    }
    "LUL";
    {
        "displayName";
        "Luxemburgischer Finanz-Franc",
        "displayName-count-one";
        "Luxemburgische Finanz-Franc",
        "displayName-count-other";
        "Luxemburgische Finanz-Franc",
        "symbol";
        "LUL";
    }
    "LVL";
    {
        "displayName";
        "Lettischer Lats",
        "displayName-count-one";
        "Lettischer Lats",

```

```

        "displayName-count-other";
        "Lettische Lats",
        "symbol";
        "LVL",
        "symbol-alt-narrow";
        "Ls";
    }
    "LVR";
    {
        "displayName";
        "Lettischer Rubel",
        "displayName-count-one";
        "Lettische Rubel",
        "displayName-count-other";
        "Lettische Rubel",
        "symbol";
        "LVR";
    }
    "LYD";
    {
        "displayName";
        "Libyscher Dinar",
        "displayName-count-one";
        "Libyscher Dinar",
        "displayName-count-other";
        "Libysche Dinar",
        "symbol";
        "LYD";
    }
    "MAD";
    {
        "displayName";
        "Marokkanischer Dirham",
        "displayName-count-one";
        "Marokkanischer Dirham",
        "displayName-count-other";
        "Marokkanische Dirham",
        "symbol";
        "MAD";
    }
    "MAF";
    {
        "displayName";
        "Marokkanischer Franc",
        "displayName-count-one";
        "Marokkanische Franc",
        "displayName-count-other";
        "Marokkanische Franc",
        "symbol";
        "MAF";
    }
    "MCF";
    {
        "displayName";
        "Monegassischer Franc",
        "displayName-count-one";
        "Monegassischer Franc",

```

```

        "displayName-count-other";
        "Monegassische Franc",
        "symbol";
        "MCF";
    }
    "MDC";
    {
        "displayName";
        "Moldau-Cupon",
        "displayName-count-one";
        "Moldau-Cupon",
        "displayName-count-other";
        "Moldau-Cupon",
        "symbol";
        "MDC";
    }
    "MDL";
    {
        "displayName";
        "Moldau-Leu",
        "displayName-count-one";
        "Moldau-Leu",
        "displayName-count-other";
        "Moldau-Leu",
        "symbol";
        "MDL";
    }
    "MGA";
    {
        "displayName";
        "Madagaskar-Ariary",
        "displayName-count-one";
        "Madagaskar-Ariary",
        "displayName-count-other";
        "Madagaskar-Ariary",
        "symbol";
        "MGA",
        "symbol-alt-narrow";
        "Ar";
    }
    "MGF";
    {
        "displayName";
        "Madagaskar-Franc",
        "displayName-count-one";
        "Madagaskar-Franc",
        "displayName-count-other";
        "Madagaskar-Franc",
        "symbol";
        "MGF";
    }
    "MKD";
    {
        "displayName";
        "Mazedonischer Denar",
        "displayName-count-one";
        "Mazedonischer Denar",

```

```

        "displayName-count-other";
        "Mazedonische Denari",
        "symbol";
        "MKD";
    }
    "MKN";
    {
        "displayName";
        "Mazedonischer Denar (1992-1993)",
        "displayName-count-one";
        "Mazedonischer Denar (1992-1993)",
        "displayName-count-other";
        "Mazedonische Denar (1992-1993)",
        "symbol";
        "MKN";
    }
    "MLF";
    {
        "displayName";
        "Malischer Franc",
        "displayName-count-one";
        "Malische Franc",
        "displayName-count-other";
        "Malische Franc",
        "symbol";
        "MLF";
    }
    "MMK";
    {
        "displayName";
        "Myanmarischer Kyat",
        "displayName-count-one";
        "Myanmarischer Kyat",
        "displayName-count-other";
        "Myanmarische Kyat",
        "symbol";
        "MMK",
        "symbol-alt-narrow";
        "K";
    }
    "MNT";
    {
        "displayName";
        "Mongolischer Tögrög",
        "displayName-count-one";
        "Mongolischer Tögrög",
        "displayName-count-other";
        "Mongolische Tögrög",
        "symbol";
        "MNT",
        "symbol-alt-narrow";
        "₮";
    }
    "MOP";
    {
        "displayName";
        "Macao-Pataca",

```

```

        "displayName-count-one";
        "Macao-Pataca",
        "displayName-count-other";
        "Macao-Pataca",
        "symbol";
        "MOP";
    }
    "MRO";
    {
        "displayName";
        "Mauretanischer Ouguiya",
        "displayName-count-one";
        "Mauretanischer Ouguiya",
        "displayName-count-other";
        "Mauretansiche Ouguiya",
        "symbol";
        "MRO";
    }
    "MTL";
    {
        "displayName";
        "Maltesische Lira",
        "displayName-count-one";
        "Maltesische Lira",
        "displayName-count-other";
        "Maltesische Lira",
        "symbol";
        "MTL";
    }
    "MTP";
    {
        "displayName";
        "Maltesisches Pfund",
        "displayName-count-one";
        "Maltesische Pfund",
        "displayName-count-other";
        "Maltesische Pfund",
        "symbol";
        "MTP";
    }
    "MUR";
    {
        "displayName";
        "Mauritius-Rupie",
        "displayName-count-one";
        "Mauritius-Rupie",
        "displayName-count-other";
        "Mauritius-Rupien",
        "symbol";
        "MUR",
        "symbol-alt-narrow";
        "Rs";
    }
    "MVP";
    {
        "displayName";
        "Malediven-Rupie (alt)",

```

```

        "displayName-count-one";
        "Malediven-Rupie (alt)",
        "displayName-count-other";
        "Malediven-Rupien (alt)";
    }
    "MVR";
    {
        "displayName";
        "Malediven-Rufiyaa",
        "displayName-count-one";
        "Malediven-Rufiyaa",
        "displayName-count-other";
        "Malediven-Rupien",
        "symbol";
        "MVR";
    }
    "MWK";
    {
        "displayName";
        "Malawi-Kwacha",
        "displayName-count-one";
        "Malawi-Kwacha",
        "displayName-count-other";
        "Malawi-Kwacha",
        "symbol";
        "MWK";
    }
    "MXN";
    {
        "displayName";
        "Mexikanischer Peso",
        "displayName-count-one";
        "Mexikanischer Peso",
        "displayName-count-other";
        "Mexikanische Pesos",
        "symbol";
        "MX$",
        "symbol-alt-narrow";
        "$";
    }
    "MXP";
    {
        "displayName";
        "Mexikanischer Silber-Peso (1861-1992)",
        "displayName-count-one";
        "Mexikanische Silber-Peso (1861-1992)",
        "displayName-count-other";
        "Mexikanische Silber-Pesos (1861-1992)",
        "symbol";
        "MXP";
    }
    "MXV";
    {
        "displayName";
        "Mexicanischer Unidad de Inversion (UDI)",
        "displayName-count-one";
        "Mexicanischer Unidad de Inversion (UDI)",

```



```

        "displayName-count-other";
        "Mexikanische Unidad de Inversion (UDI)",
        "symbol";
        "MXV";
    }
    "MYR";
    {
        "displayName";
        "Malaysischer Ringgit",
        "displayName-count-one";
        "Malaysischer Ringgit",
        "displayName-count-other";
        "Malaysische Ringgit",
        "symbol";
        "MYR",
        "symbol-alt-narrow";
        "RM";
    }
    "MZE";
    {
        "displayName";
        "Mosambikanischer Escudo",
        "displayName-count-one";
        "Mozambikanische Escudo",
        "displayName-count-other";
        "Mozambikanische Escudo",
        "symbol";
        "MZE";
    }
    "MZM";
    {
        "displayName";
        "Mosambikanischer Metical (1980-2006)",
        "displayName-count-one";
        "Mosambikanischer Metical (1980-2006)",
        "displayName-count-other";
        "Mosambikanische Meticais (1980-2006)",
        "symbol";
        "MZM";
    }
    "MZN";
    {
        "displayName";
        "Mosambikanischer Metical",
        "displayName-count-one";
        "Mosambikanischer Metical",
        "displayName-count-other";
        "Mosambikanische Meticais",
        "symbol";
        "MZN";
    }
    "NAD";
    {
        "displayName";
        "Namibia-Dollar",
        "displayName-count-one";
        "Namibia-Dollar",

```

```

        "displayName-count-other";
        "Namibia-Dollar",
        "symbol";
        "NAD",
        "symbol-alt-narrow";
        "$";
    }
    "NGN";
    {
        "displayName";
        "Nigerianischer Naira",
        "displayName-count-one";
        "Nigerianischer Naira",
        "displayName-count-other";
        "Nigerianische Naira",
        "symbol";
        "NGN",
        "symbol-alt-narrow";
        "₦";
    }
    "NIC";
    {
        "displayName";
        "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-one";
        "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-other";
        "Nicaraguanische Córdoba (1988-1991)",
        "symbol";
        "NIC";
    }
    "NIO";
    {
        "displayName";
        "Nicaragua-Córdoba",
        "displayName-count-one";
        "Nicaragua-Córdoba",
        "displayName-count-other";
        "Nicaragua-Córdobas",
        "symbol";
        "NIO",
        "symbol-alt-narrow";
        "C$";
    }
    "NLG";
    {
        "displayName";
        "Niederländischer Gulden",
        "displayName-count-one";
        "Niederländischer Gulden",
        "displayName-count-other";
        "Niederländische Gulden",
        "symbol";
        "NLG";
    }
    "NOK";
    {

```

```

        "displayName";
        "Norwegische Krone",
        "displayName-count-one";
        "Norwegische Krone",
        "displayName-count-other";
        "Norwegische Kronen",
        "symbol";
        "NOK",
        "symbol-alt-narrow";
        "kr";
    }
    "NPR";
    {
        "displayName";
        "Nepalesische Rupie",
        "displayName-count-one";
        "Nepalesische Rupie",
        "displayName-count-other";
        "Nepalesische Rupien",
        "symbol";
        "NPR",
        "symbol-alt-narrow";
        "Rs";
    }
    "NZD";
    {
        "displayName";
        "Neuseeland-Dollar",
        "displayName-count-one";
        "Neuseeland-Dollar",
        "displayName-count-other";
        "Neuseeland-Dollar",
        "symbol";
        "NZ$",
        "symbol-alt-narrow";
        "$";
    }
    "OMR";
    {
        "displayName";
        "Omanischer Rial",
        "displayName-count-one";
        "Omanischer Rial",
        "displayName-count-other";
        "Omanische Rials",
        "symbol";
        "OMR";
    }
    "PAB";
    {
        "displayName";
        "Panamaischer Balboa",
        "displayName-count-one";
        "Panamaischer Balboa",
        "displayName-count-other";
        "Panamaische Balboas",
        "symbol";
    }

```

```

        "PAB";
    }
    "PEI";
    {
        "displayName";
        "Peruanischer Inti",
        "displayName-count-one";
        "Peruanische Inti",
        "displayName-count-other";
        "Peruanische Inti",
        "symbol";
        "PEI";
    }
    "PEN";
    {
        "displayName";
        "Peruanischer Sol",
        "displayName-count-one";
        "Peruanischer Sol",
        "displayName-count-other";
        "Peruanische Sol",
        "symbol";
        "PEN";
    }
    "PES";
    {
        "displayName";
        "Peruanischer Sol (1863-1965)",
        "displayName-count-one";
        "Peruanischer Sol (1863-1965)",
        "displayName-count-other";
        "Peruanische Sol (1863-1965)",
        "symbol";
        "PES";
    }
    "PGK";
    {
        "displayName";
        "Papua-Neuguineischer Kina",
        "displayName-count-one";
        "Papua-Neuguineischer Kina",
        "displayName-count-other";
        "Papua-Neuguineische Kina",
        "symbol";
        "PGK";
    }
    "PHP";
    {
        "displayName";
        "Philippinischer Peso",
        "displayName-count-one";
        "Philippinischer Peso",
        "displayName-count-other";
        "Philippinische Pesos",
        "symbol";
        "PHP",
        "symbol-alt-narrow";
    }

```

```

        "₹";
    }
    "PKR";
    {
        "displayName";
        "Pakistanische Rupie",
        "displayName-count-one";
        "Pakistanische Rupie",
        "displayName-count-other";
        "Pakistanische Rupien",
        "symbol";
        "PKR",
        "symbol-alt-narrow";
        "Rs";
    }
    "PLN";
    {
        "displayName";
        "Polnischer Złoty",
        "displayName-count-one";
        "Polnischer Złoty",
        "displayName-count-other";
        "Polnische Złoty",
        "symbol";
        "PLN",
        "symbol-alt-narrow";
        "zł";
    }
    "PLZ";
    {
        "displayName";
        "Polnischer Zloty (1950-1995)",
        "displayName-count-one";
        "Polnischer Zloty (1950-1995)",
        "displayName-count-other";
        "Polnische Zloty (1950-1995)",
        "symbol";
        "PLZ";
    }
    "PTE";
    {
        "displayName";
        "Portugiesischer Escudo",
        "displayName-count-one";
        "Portugiesische Escudo",
        "displayName-count-other";
        "Portugiesische Escudo",
        "symbol";
        "PTE";
    }
    "PYG";
    {
        "displayName";
        "Paraguayischer Guaraní",
        "displayName-count-one";
        "Paraguayischer Guaraní",
        "displayName-count-other";
    }

```

```

        "Paraguayische Guaraníes",
        "symbol";
    "PYG",
        "symbol-alt-narrow";
    "₲";
}
"QAR";
{
    "displayName";
    "Katar-Riyal",
        "displayName-count-one";
    "Katar-Riyal",
        "displayName-count-other";
    "Katar-Riyal",
        "symbol";
    "QAR";
}
"RHD";
{
    "displayName";
    "Rhodesischer Dollar",
        "displayName-count-one";
    "Rhodesische Dollar",
        "displayName-count-other";
    "Rhodesische Dollar",
        "symbol";
    "RHD";
}
"ROL";
{
    "displayName";
    "Rumänischer Leu (1952-2006)",
        "displayName-count-one";
    "Rumänischer Leu (1952-2006)",
        "displayName-count-other";
    "Rumänische Leu (1952-2006)",
        "symbol";
    "ROL";
}
"RON";
{
    "displayName";
    "Rumänischer Leu",
        "displayName-count-one";
    "Rumänischer Leu",
        "displayName-count-other";
    "Rumänische Leu",
        "symbol";
    "RON",
        "symbol-alt-narrow";
    "L";
}
"RSD";
{
    "displayName";
    "Serbischer Dinar",
        "displayName-count-one";

```

```

        "Serbischer Dinar",
        "displayName-count-other";
    "Serbische Dinaren",
        "symbol";
    "RSD";
}
"RUB";
{
    "displayName";
    "Russischer Rubel",
        "displayName-count-one";
    "Russischer Rubel",
        "displayName-count-other";
    "Russische Rubel",
        "symbol";
    "RUB",
        "symbol-alt-narrow";
    "₽";
}
"RUR";
{
    "displayName";
    "Russischer Rubel (1991-1998)",
        "displayName-count-one";
    "Russischer Rubel (1991-1998)",
        "displayName-count-other";
    "Russische Rubel (1991-1998)",
        "symbol";
    "RUR",
        "symbol-alt-narrow";
    "p.";
}
"RWF";
{
    "displayName";
    "Ruanda-Franc",
        "displayName-count-one";
    "Ruanda-Franc",
        "displayName-count-other";
    "Ruanda-Francs",
        "symbol";
    "RWF",
        "symbol-alt-narrow";
    "F.Rw";
}
"SAR";
{
    "displayName";
    "Saudi-Rial",
        "displayName-count-one";
    "Saudi-Rial",
        "displayName-count-other";
    "Saudi-Rial",
        "symbol";
    "SAR";
}
"SBD";

```

```

    {
      "displayName";
      "Salomonen-Dollar",
      "displayName-count-one";
      "Salomonen-Dollar",
      "displayName-count-other";
      "Salomonen-Dollar",
      "symbol";
      "SBD",
      "symbol-alt-narrow";
      "$";
    }
    "SCR";
    {
      "displayName";
      "Seychellen-Rupie",
      "displayName-count-one";
      "Seychellen-Rupie",
      "displayName-count-other";
      "Seychellen-Rupien",
      "symbol";
      "SCR";
    }
    "SDD";
    {
      "displayName";
      "Sudanesischer Dinar (1992-2007)",
      "displayName-count-one";
      "Sudanesischer Dinar (1992-2007)",
      "displayName-count-other";
      "Sudanesische Dinar (1992-2007)",
      "symbol";
      "SDD";
    }
    "SDG";
    {
      "displayName";
      "Sudanesisches Pfund",
      "displayName-count-one";
      "Sudanesisches Pfund",
      "displayName-count-other";
      "Sudanesische Pfund",
      "symbol";
      "SDG";
    }
    "SDP";
    {
      "displayName";
      "Sudanesisches Pfund (1957-1998)",
      "displayName-count-one";
      "Sudanesisches Pfund (1957-1998)",
      "displayName-count-other";
      "Sudanesische Pfund (1957-1998)",
      "symbol";
      "SDP";
    }
    "SEK";

```



```

    {
      "displayName";
      "Schwedische Krone",
        "displayName-count-one";
      "Schwedische Krone",
        "displayName-count-other";
      "Schwedische Kronen",
        "symbol";
      "SEK",
        "symbol-alt-narrow";
      "kr";
    }
    "SGD";
    {
      "displayName";
      "Singapur-Dollar",
        "displayName-count-one";
      "Singapur-Dollar",
        "displayName-count-other";
      "Singapur-Dollar",
        "symbol";
      "SGD",
        "symbol-alt-narrow";
      "$";
    }
    "SHP";
    {
      "displayName";
      "St. Helena-Pfund",
        "displayName-count-one";
      "St. Helena-Pfund",
        "displayName-count-other";
      "St. Helena-Pfund",
        "symbol";
      "SHP",
        "symbol-alt-narrow";
      "£";
    }
    "SIT";
    {
      "displayName";
      "Slowenischer Tolar",
        "displayName-count-one";
      "Slowenischer Tolar",
        "displayName-count-other";
      "Slowenische Tolar",
        "symbol";
      "SIT";
    }
    "SKK";
    {
      "displayName";
      "Slowakische Krone",
        "displayName-count-one";
      "Slowakische Kronen",
        "displayName-count-other";
      "Slowakische Kronen",

```

```

        "symbol";
        "SKK";
    }
    "SLL";
    {
        "displayName";
        "Sierra-leonischer Leone",
        "displayName-count-one";
        "Sierra-leonischer Leone",
        "displayName-count-other";
        "Sierra-leonische Leones",
        "symbol";
        "SLL";
    }
    "SOS";
    {
        "displayName";
        "Somalia-Schilling",
        "displayName-count-one";
        "Somalia-Schilling",
        "displayName-count-other";
        "Somalia-Schilling",
        "symbol";
        "SOS";
    }
    "SRD";
    {
        "displayName";
        "Suriname-Dollar",
        "displayName-count-one";
        "Suriname-Dollar",
        "displayName-count-other";
        "Suriname-Dollar",
        "symbol";
        "SRD",
        "symbol-alt-narrow";
        "$";
    }
    "SRG";
    {
        "displayName";
        "Suriname Gulden",
        "displayName-count-one";
        "Suriname-Gulden",
        "displayName-count-other";
        "Suriname-Gulden",
        "symbol";
        "SRG";
    }
    "SSP";
    {
        "displayName";
        "Südsudanesisches Pfund",
        "displayName-count-one";
        "Südsudanesisches Pfund",
        "displayName-count-other";
        "Südsudanesische Pfund",

```

```

        "symbol";
        "SSP",
        "symbol-alt-narrow";
        "£";
    }
    "STD";
    {
        "displayName";
        "São-toméischer Dobra",
        "displayName-count-one";
        "São-toméischer Dobra",
        "displayName-count-other";
        "São-toméische Dobra",
        "symbol";
        "STD",
        "symbol-alt-narrow";
        "Db";
    }
    "SUR";
    {
        "displayName";
        "Sowjetischer Rubel",
        "displayName-count-one";
        "Sowjetische Rubel",
        "displayName-count-other";
        "Sowjetische Rubel",
        "symbol";
        "SUR";
    }
    "SVC";
    {
        "displayName";
        "El Salvador Colon",
        "displayName-count-one";
        "El Salvador-Colon",
        "displayName-count-other";
        "El Salvador-Colon",
        "symbol";
        "SVC";
    }
    "SYP";
    {
        "displayName";
        "Syrisches Pfund",
        "displayName-count-one";
        "Syrisches Pfund",
        "displayName-count-other";
        "Syrische Pfund",
        "symbol";
        "SYP",
        "symbol-alt-narrow";
        "SYP";
    }
    "SZL";
    {
        "displayName";
        "Swasiländischer Lilangeni",

```

```

        "displayName-count-one";
        "Swasiländischer Lilangeni",
        "displayName-count-other";
        "Swasiländische Emalangeni",
        "symbol";
        "SZL";
    }
    "THB";
    {
        "displayName";
        "Thailändischer Baht",
        "displayName-count-one";
        "Thailändischer Baht",
        "displayName-count-other";
        "Thailändische Baht",
        "symbol";
        "฿",
        "symbol-alt-narrow";
        "฿";
    }
    "TJR";
    {
        "displayName";
        "Tadschikistan Rubel",
        "displayName-count-one";
        "Tadschikistan-Rubel",
        "displayName-count-other";
        "Tadschikistan-Rubel",
        "symbol";
        "TJR";
    }
    "TJS";
    {
        "displayName";
        "Tadschikistan-Somoni",
        "displayName-count-one";
        "Tadschikistan-Somoni",
        "displayName-count-other";
        "Tadschikistan-Somoni",
        "symbol";
        "TJS";
    }
    "TMM";
    {
        "displayName";
        "Turkmenistan-Manat (1993-2009)",
        "displayName-count-one";
        "Turkmenistan-Manat (1993-2009)",
        "displayName-count-other";
        "Turkmenistan-Manat (1993-2009)",
        "symbol";
        "TMM";
    }
    "TMT";
    {
        "displayName";
        "Turkmenistan-Manat",

```

```

        "displayName-count-one";
        "Turkmenistan-Manat",
        "displayName-count-other";
        "Turkmenistan-Manat",
        "symbol";
        "TMT";
    }
    "TND";
    {
        "displayName";
        "Tunesischer Dinar",
        "displayName-count-one";
        "Tunesischer Dinar",
        "displayName-count-other";
        "Tunesische Dinar",
        "symbol";
        "TND";
    }
    "TOP";
    {
        "displayName";
        "Tongaischer Pa'anga",
        "displayName-count-one";
        "Tongaischer Pa'anga",
        "displayName-count-other";
        "Tongaische Pa'anga",
        "symbol";
        "TOP",
        "symbol-alt-narrow";
        "T$";
    }
    "TPE";
    {
        "displayName";
        "Timor-Escudo",
        "displayName-count-one";
        "Timor-Escudo",
        "displayName-count-other";
        "Timor-Escudo",
        "symbol";
        "TPE";
    }
    "TRL";
    {
        "displayName";
        "Türkische Lira (1922-2005)",
        "displayName-count-one";
        "Türkische Lira (1922-2005)",
        "displayName-count-other";
        "Türkische Lira (1922-2005)",
        "symbol";
        "TRL";
    }
    "TRY";
    {
        "displayName";
        "Türkische Lira",

```

```

        "displayName-count-one";
        "Türkische Lira",
        "displayName-count-other";
        "Türkische Lira",
        "symbol";
        "TRY",
        "symbol-alt-narrow";
        "₺",
        "symbol-alt-variant";
        "TL";
    }
    "TTD";
    {
        "displayName";
        "Trinidad und Tobago-Dollar",
        "displayName-count-one";
        "Trinidad und Tobago-Dollar",
        "displayName-count-other";
        "Trinidad und Tobago-Dollar",
        "symbol";
        "TTD",
        "symbol-alt-narrow";
        "$";
    }
    "TWD";
    {
        "displayName";
        "Neuer Taiwan-Dollar",
        "displayName-count-one";
        "Neuer Taiwan-Dollar",
        "displayName-count-other";
        "Neue Taiwan-Dollar",
        "symbol";
        "NT$",
        "symbol-alt-narrow";
        "NT$";
    }
    "TZS";
    {
        "displayName";
        "Tansania-Schilling",
        "displayName-count-one";
        "Tansania-Schilling",
        "displayName-count-other";
        "Tansania-Schilling",
        "symbol";
        "TZS";
    }
    "UAH";
    {
        "displayName";
        "Ukrainische Hrywnja",
        "displayName-count-one";
        "Ukrainische Hrywnja",
        "displayName-count-other";
        "Ukrainische Hrywen",
        "symbol";
    }

```

```

        "UAH",
        "symbol-alt-narrow";
        "₴";
    }
    "UAK";
    {
        "displayName";
        "Ukrainischer Karbovanetz",
        "displayName-count-one";
        "Ukrainische Karbovanetz",
        "displayName-count-other";
        "Ukrainische Karbovanetz",
        "symbol";
        "UAK";
    }
    "UGS";
    {
        "displayName";
        "Uganda-Schilling (1966-1987)",
        "displayName-count-one";
        "Uganda-Schilling (1966-1987)",
        "displayName-count-other";
        "Uganda-Schilling (1966-1987)",
        "symbol";
        "UGS";
    }
    "UGX";
    {
        "displayName";
        "Uganda-Schilling",
        "displayName-count-one";
        "Uganda-Schilling",
        "displayName-count-other";
        "Uganda-Schilling",
        "symbol";
        "UGX";
    }
    "USD";
    {
        "displayName";
        "US-Dollar",
        "displayName-count-one";
        "US-Dollar",
        "displayName-count-other";
        "US-Dollar",
        "symbol";
        "$",
        "symbol-alt-narrow";
        "$";
    }
    "USN";
    {
        "displayName";
        "US Dollar (Nächster Tag)",
        "displayName-count-one";
        "US-Dollar (Nächster Tag)",
        "displayName-count-other";
    }

```

```

        "US-Dollar (Nächster Tag)",
        "symbol";
        "USN";
    }
    "USS";
    {
        "displayName";
        "US Dollar (Gleicher Tag)",
        "displayName-count-one";
        "US-Dollar (Gleicher Tag)",
        "displayName-count-other";
        "US-Dollar (Gleicher Tag)",
        "symbol";
        "USS";
    }
    "UYI";
    {
        "displayName";
        "Uruguayischer Peso (Indexierte Rechnungseinheiten)",
        "displayName-count-one";
        "Uruguayischer Peso (Indexierte Rechnungseinheiten)",
        "displayName-count-other";
        "Uruguayische Pesos (Indexierte Rechnungseinheiten)",
        "symbol";
        "UYI";
    }
    "UYP";
    {
        "displayName";
        "Uruguayischer Peso (1975-1993)",
        "displayName-count-one";
        "Uruguayischer Peso (1975-1993)",
        "displayName-count-other";
        "Uruguayische Pesos (1975-1993)",
        "symbol";
        "UYP";
    }
    "UYU";
    {
        "displayName";
        "Uruguayischer Peso",
        "displayName-count-one";
        "Uruguayischer Peso",
        "displayName-count-other";
        "Uruguayische Pesos",
        "symbol";
        "UYU",
        "symbol-alt-narrow";
        "$";
    }
    "UZS";
    {
        "displayName";
        "Usbekistan-Sum",
        "displayName-count-one";
        "Usbekistan-Sum",
        "displayName-count-other";
    }

```



```

        "Usbekistan-Sum",
        "symbol";
        "UZS";
    }
    "VEB";
    {
        "displayName";
        "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-one";
        "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-other";
        "Venezolanische Bolívares (1871-2008)",
        "symbol";
        "VEB";
    }
    "VEF";
    {
        "displayName";
        "Venezolanischer Bolívar",
        "displayName-count-one";
        "Venezolanischer Bolívar",
        "displayName-count-other";
        "Venezolanische Bolívares",
        "symbol";
        "VEF",
        "symbol-alt-narrow";
        "Bs";
    }
    "VND";
    {
        "displayName";
        "Vietnamesischer Dong",
        "displayName-count-one";
        "Vietnamesischer Dong",
        "displayName-count-other";
        "Vietnamesische Dong",
        "symbol";
        "₫",
        "symbol-alt-narrow";
        "₫";
    }
    "VNN";
    {
        "displayName";
        "Vietnamesischer Dong (1978-1985)",
        "displayName-count-one";
        "Vietnamesischer Dong (1978-1985)",
        "displayName-count-other";
        "Vietnamesische Dong (1978-1985)",
        "symbol";
        "VNN";
    }
    "VUV";
    {
        "displayName";
        "Vanuatu-Vatu",
        "displayName-count-one";

```

```

        "Vanuatu-Vatu",
        "displayName-count-other";
        "Vanuatu-Vatu",
        "symbol";
        "VUV";
    }
    "WST";
    {
        "displayName";
        "Samoanischer Tala",
        "displayName-count-one";
        "Samoanischer Tala",
        "displayName-count-other";
        "Samoanische Tala",
        "symbol";
        "WST";
    }
    "XAF";
    {
        "displayName";
        "CFA-Franc (BEAC) ",
        "displayName-count-one";
        "CFA-Franc (BEAC) ",
        "displayName-count-other";
        "CFA-Franc (BEAC) ",
        "symbol";
        "FCFA";
    }
    "XAG";
    {
        "displayName";
        "Unze Silber",
        "displayName-count-one";
        "Unze Silber",
        "displayName-count-other";
        "Unzen Silber",
        "symbol";
        "XAG";
    }
    "XAU";
    {
        "displayName";
        "Unze Gold",
        "displayName-count-one";
        "Unze Gold",
        "displayName-count-other";
        "Unzen Gold",
        "symbol";
        "XAU";
    }
    "XBA";
    {
        "displayName";
        "Europäische Rechnungseinheit",
        "displayName-count-one";
        "Europäische Rechnungseinheiten",
        "displayName-count-other";
    }

```

```

        "Europäische Rechnungseinheiten",
        "symbol";
        "XBA";
    }
    "XBB";
    {
        "displayName";
        "Europäische Währungseinheit (XBB)",
        "displayName-count-one";
        "Europäische Währungseinheiten (XBB)",
        "displayName-count-other";
        "Europäische Währungseinheiten (XBB)",
        "symbol";
        "XBB";
    }
    "XBC";
    {
        "displayName";
        "Europäische Rechnungseinheit (XBC)",
        "displayName-count-one";
        "Europäische Rechnungseinheiten (XBC)",
        "displayName-count-other";
        "Europäische Rechnungseinheiten (XBC)",
        "symbol";
        "XBC";
    }
    "XBD";
    {
        "displayName";
        "Europäische Rechnungseinheit (XBD)",
        "displayName-count-one";
        "Europäische Rechnungseinheiten (XBD)",
        "displayName-count-other";
        "Europäische Rechnungseinheiten (XBD)",
        "symbol";
        "XBD";
    }
    "XCD";
    {
        "displayName";
        "Ostkaribischer Dollar",
        "displayName-count-one";
        "Ostkaribischer Dollar",
        "displayName-count-other";
        "Ostkaribische Dollar",
        "symbol";
        "EC$",
        "symbol-alt-narrow";
        "$";
    }
    "XDR";
    {
        "displayName";
        "Sonderziehungsrechte",
        "displayName-count-one";
        "Sonderziehungsrechte",
        "displayName-count-other";
    }

```

```

        "Sonderziehungsrechte",
        "symbol";
        "XDR";
    }
    "XEU";
    {
        "displayName";
        "Europäische Währungseinheit (XEU)",
        "displayName-count-one";
        "Europäische Währungseinheiten (XEU)",
        "displayName-count-other";
        "Europäische Währungseinheiten (XEU)",
        "symbol";
        "XEU";
    }
    "XFO";
    {
        "displayName";
        "Französischer Gold-Franc",
        "displayName-count-one";
        "Französische Gold-Franc",
        "displayName-count-other";
        "Französische Gold-Franc",
        "symbol";
        "XFO";
    }
    "XFU";
    {
        "displayName";
        "Französischer UIC-Franc",
        "displayName-count-one";
        "Französische UIC-Franc",
        "displayName-count-other";
        "Französische UIC-Franc",
        "symbol";
        "XFU";
    }
    "XOF";
    {
        "displayName";
        "CFA-Franc (BCEAO)",
        "displayName-count-one";
        "CFA-Franc (BCEAO)",
        "displayName-count-other";
        "CFA-Francs (BCEAO)",
        "symbol";
        "CFA";
    }
    "XPD";
    {
        "displayName";
        "Unze Palladium",
        "displayName-count-one";
        "Unze Palladium",
        "displayName-count-other";
        "Unzen Palladium",
        "symbol";
    }

```

```

        "XPD";
    }
    "XPF";
    {
        "displayName";
        "CFP-Franc",
            "displayName-count-one";
        "CFP-Franc",
            "displayName-count-other";
        "CFP-Franc",
            "symbol";
        "CFPF";
    }
    "XPT";
    {
        "displayName";
        "Unze Platin",
            "displayName-count-one";
        "Unze Platin",
            "displayName-count-other";
        "Unzen Platin",
            "symbol";
        "XPT";
    }
    "XRE";
    {
        "displayName";
        "RINET Funds",
            "displayName-count-one";
        "RINET Funds",
            "displayName-count-other";
        "RINET Funds",
            "symbol";
        "XRE";
    }
    "XSU";
    {
        "displayName";
        "SUCRE",
            "displayName-count-one";
        "SUCRE",
            "displayName-count-other";
        "SUCRE",
            "symbol";
        "XSU";
    }
    "XTS";
    {
        "displayName";
        "Testwährung",
            "displayName-count-one";
        "Testwährung",
            "displayName-count-other";
        "Testwährung",
            "symbol";
        "XTS";
    }
}

```

```

    "XUA";
    {
        "displayName";
        "Rechnungseinheit der AfEB",
        "displayName-count-one";
        "Rechnungseinheit der AfEB",
        "displayName-count-other";
        "Rechnungseinheiten der AfEB",
        "symbol";
        "XUA";
    }
    "XXX";
    {
        "displayName";
        "Unbekannte Währung",
        "displayName-count-one";
        "(unbekannte Währung)",
        "displayName-count-other";
        "(unbekannte Währung)",
        "symbol";
        "XXX";
    }
    "YDD";
    {
        "displayName";
        "Jemen-Dinar",
        "displayName-count-one";
        "Jemen-Dinar",
        "displayName-count-other";
        "Jemen-Dinar",
        "symbol";
        "YDD";
    }
    "YER";
    {
        "displayName";
        "Jemen-Rial",
        "displayName-count-one";
        "Jemen-Rial",
        "displayName-count-other";
        "Jemen-Rial",
        "symbol";
        "YER";
    }
    "YUD";
    {
        "displayName";
        "Jugoslawischer Dinar (1966-1990)",
        "displayName-count-one";
        "Jugoslawischer Dinar (1966-1990)",
        "displayName-count-other";
        "Jugoslawische Dinar (1966-1990)",
        "symbol";
        "YUD";
    }
    "YUM";
    {

```

```

        "displayName";
        "Jugoslawischer Neuer Dinar (1994-2002)",
        "displayName-count-one";
        "Jugoslawischer Neuer Dinar (1994-2002)",
        "displayName-count-other";
        "Jugoslawische Neue Dinar (1994-2002)",
        "symbol";
        "YUM";
    }
    "YUN";
    {
        "displayName";
        "Jugoslawischer Dinar (konvertibel)",
        "displayName-count-one";
        "Jugoslawische Dinar (konvertibel)",
        "displayName-count-other";
        "Jugoslawische Dinar (konvertibel)",
        "symbol";
        "YUN";
    }
    "YUR";
    {
        "displayName";
        "Jugoslawischer reformierter Dinar (1992-1993)",
        "displayName-count-one";
        "Jugoslawischer reformierter Dinar (1992-1993)",
        "displayName-count-other";
        "Jugoslawische reformierte Dinar (1992-1993)",
        "symbol";
        "YUR";
    }
    "ZAL";
    {
        "displayName";
        "Südafrikanischer Rand (Finanz)",
        "displayName-count-one";
        "Südafrikanischer Rand (Finanz)",
        "displayName-count-other";
        "Südafrikanischer Rand (Finanz)",
        "symbol";
        "ZAL";
    }
    "ZAR";
    {
        "displayName";
        "Südafrikanischer Rand",
        "displayName-count-one";
        "Südafrikanischer Rand",
        "displayName-count-other";
        "Südafrikanische Rand",
        "symbol";
        "ZAR",
        "symbol-alt-narrow";
        "R";
    }
    "ZMK";
    {

```

```

        "displayName";
        "Kwacha (1968-2012)",
        "displayName-count-one";
        "Kwacha (1968-2012)",
        "displayName-count-other";
        "Kwacha (1968-2012)",
        "symbol";
        "ZMK";
    }
    "ZMW";
    {
        "displayName";
        "Kwacha",
        "displayName-count-one";
        "Kwacha",
        "displayName-count-other";
        "Kwacha",
        "symbol";
        "ZMW",
        "symbol-alt-narrow";
        "K";
    }
    "ZRN";
    {
        "displayName";
        "Zaire-Neuer Zaïre (1993-1998)",
        "displayName-count-one";
        "Zaire-Neuer Zaïre (1993-1998)",
        "displayName-count-other";
        "Zaire-Neue Zaïre (1993-1998)",
        "symbol";
        "ZRN";
    }
    "ZRZ";
    {
        "displayName";
        "Zaire-Zaïre (1971-1993)",
        "displayName-count-one";
        "Zaire-Zaïre (1971-1993)",
        "displayName-count-other";
        "Zaire-Zaïre (1971-1993)",
        "symbol";
        "ZRZ";
    }
    "ZWD";
    {
        "displayName";
        "Simbabwe-Dollar (1980-2008)",
        "displayName-count-one";
        "Simbabwe-Dollar (1980-2008)",
        "displayName-count-other";
        "Simbabwe-Dollar (1980-2008)",
        "symbol";
        "ZWD";
    }
    "ZWL";
    {

```



```
"displayName";
"Simbabwe-Dollar (2009)",
    "displayName-count-one";
"Simbabwe-Dollar (2009)",
    "displayName-count-other";
"Simbabwe-Dollar (2009)",
    "symbol";
"ZWL";
}
"ZWR";
{
    "displayName";
    "Simbabwe-Dollar (2008)",
        "displayName-count-one";
    "Simbabwe-Dollar (2008)",
        "displayName-count-other";
    "Simbabwe-Dollar (2008)",
        "symbol";
    "ZWR";
}
}
}
}
```

INDEX.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// Load the L10n, loadCldr from ej2-base
import { loadCldr, L10n } from '@syncfusion/ej2-base';
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
// load the CLDR data files.
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as timeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames);
L10n.load({
    'de': {
        'datepicker': { placeholder: 'Wählen Sie ein Datum aus',
            today: 'heute' }
    }
});
// import the datepickercomponent
function App() {
    return <DatePickerComponent id="datepicker" locale='de'/>;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
```

```
// Load the L10n, loadCldr from ej2-base
import { loadCldr, L10n } from '@syncfusion/ej2-base';
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
// load the CLDR data files.
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as timeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames);
L10n.load({
    'de': {
        'datepicker': { placeholder: 'Wählen Sie ein Datum aus',
            today: 'heute' }
    }
});
// import the datepickercomponent
function App() {
    return <DatePickerComponent id="datepicker" locale='de' />;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

NUMBERINGSYSTEMS.JSON

```
{
  "supplemental": {
    "version": {
      "_number": "$Revision: 12732 $",
      "_unicodeVersion": "9.0.0",
      "_cldrVersion": "31"
    },
    "numberingSystems": {
      "adlm": {
        "_digits": "ᲠᲡᲢᲣᲤᲥᲦᲧᲨᲩ",
        "_type": "numeric"
      },
      "ahom": {
        "_digits": "᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍ",
        "_type": "numeric"
      },
      "arab": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "arabext": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "armn": {
        "_rules": "armenian-upper",
        "_type": "algorithmic"
      },
      "armnlow": {
        "_rules": "armenian-lower",
        "_type": "algorithmic"
      }
    }
  },
}
```

```
"bali": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"beng": {
  "_digits": "০১২৩৪৫৬৭৮৯",
  "_type": "numeric"
},
"bhks": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"brah": {
  "_digits": "·\\۰۱۲۳۴۵۶۷۸۹",
  "_type": "numeric"
},
"cakm": {
  "_digits": "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
  "_type": "numeric"
},
"cham": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"cyrl": {
  "_rules": "cyrillic-lower",
  "_type": "algorithmic"
},
"deva": {
  "_digits": "०१२३४५६७८९",
  "_type": "numeric"
},
"ethi": {
  "_rules": "ethiopic",
  "_type": "algorithmic"
},
"fullwide": {
  "_digits": "0 1 2 3 4 5 6 7 8 9",
  "_type": "numeric"
},
"geor": {
  "_rules": "georgian",
  "_type": "algorithmic"
},
"grek": {
  "_rules": "greek-upper",
  "_type": "algorithmic"
},
"greklow": {
  "_rules": "greek-lower",
  "_type": "algorithmic"
},
"gujr": {
  "_digits": "૦૧૨૩૪૫૬૭૮૯",
  "_type": "numeric"
}
```

```

    },
    "guru": {
      "_digits": "୦୧୨୩୪୫୬୭୮୯",
      "_type": "numeric"
    },
    "hanidays": {
      "_rules": "zh/SpelloutRules/spellout-numbering-days",
      "_type": "algorithmic"
    },
    "hanidec": {
      "_digits": "〇一二三四五六七八九",
      "_type": "numeric"
    },
    "hans": {
      "_rules": "zh/SpelloutRules/spellout-cardinal",
      "_type": "algorithmic"
    },
    "hansfin": {
      "_rules": "zh/SpelloutRules/spellout-cardinal-financial",
      "_type": "algorithmic"
    },
    "hant": {
      "_rules": "zh_Hant/SpelloutRules/spellout-cardinal",
      "_type": "algorithmic"
    },
    "hantfin": {
      "_rules": "zh_Hant/SpelloutRules/spellout-cardinal-financial",
      "_type": "algorithmic"
    },
    "hebr": {
      "_rules": "hebrew",
      "_type": "algorithmic"
    },
    "hmng": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    "java": {
      "_digits": "୦୧୨୩୪୫୬୭୮୯୦୧୨୩୪୫୬୭୮୯",
      "_type": "numeric"
    },
    "jpan": {
      "_rules": "ja/SpelloutRules/spellout-cardinal",
      "_type": "algorithmic"
    },
    "jpanfin": {
      "_rules": "ja/SpelloutRules/spellout-cardinal-financial",
      "_type": "algorithmic"
    },
    "kali": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    "khmr": {
      "_digits": "០១២៣៤៥៦៧៨៩",

```

```
    "_type": "numeric"
  },
  "knda": {
    "_digits": "೦೧೨೩೪೫೬೭೮೯",
    "_type": "numeric"
  },
  "lana": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "lanatham": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "laoo": {
    "_digits": "໐໑໒໓໔໕໖໗໘໑",
    "_type": "numeric"
  },
  "latn": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "lepc": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "limb": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "mathbold": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mathdbl": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mathmono": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mathsanb": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mathsans": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mlym": {
    "_digits": "൦൧൨൩൪൫൬൭൮൯",
    "_type": "numeric"
  },
  "modi": {
```

```

    "_digits": "□□□□□□□□",
    "_type": "numeric"
  },
  "mong": {
    "_digits": "᠐᠑᠒᠓᠔᠕᠖᠗᠘᠐",
    "_type": "numeric"
  },
  "mroo": {
    "_digits": "□□□□□□□□",
    "_type": "numeric"
  },
  "mtei": {
    "_digits": "᠐᠑ᠶ᠋ᠸᠹᠺᠻᠼᠽᠾ",
    "_type": "numeric"
  },
  "mymr": {
    "_digits": "၀၁၂၃၄၅၆၇၈",
    "_type": "numeric"
  },
  "mymrshan": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mymrtlng": {
    "_digits": "᠐ᠠᠨᠵᠢᠨᠠᠨᠢᠨᠠ",
    "_type": "numeric"
  },
  "newa": {
    "_digits": "□□□□□□□□",
    "_type": "numeric"
  },
  "nkoo": {
    "_digits": "᠐ᠠᠨᠵᠢᠨᠠᠨᠢᠨᠠ",
    "_type": "numeric"
  },
  "olck": {
    "_digits": "᠐ᠠᠨᠵᠢᠨᠠᠨᠢᠨᠠ",
    "_type": "numeric"
  },
  "orya": {
    "_digits": "୦୧୨୩୪୫୬୭୮୯",
    "_type": "numeric"
  },
  "osma": {
    "_digits": "᠐᠑ᠶ᠋ᠸᠹᠺᠻᠼᠽᠾ",
    "_type": "numeric"
  },
  "roman": {
    "_rules": "roman-upper",
    "_type": "algorithmic"
  },
  "romanlow": {
    "_rules": "roman-lower",

```

```

    "_type": "algorithmic"
  },
  "saur": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "shrd": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "sind": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "sinh": {
    "_digits": "ආශ්වතෙරභාද්‍රපදාසෘජ්‍ය",
    "_type": "numeric"
  },
  "sora": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "sund": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "takr": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "talv": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "taml": {
    "_rules": "tamil",
    "_type": "algorithmic"
  },
  "tamldec": {
    "_digits": "0௧௨௩௪௫௬௭௮௯",
    "_type": "numeric"
  },
  "telu": {
    "_digits": "0౧౨౩౪౫౬౭౮౯",
    "_type": "numeric"
  },
  "thai": {
    "_digits": "๐๑๒๓๔๕๖๗๘๙",
    "_type": "numeric"
  },
  "tibv": {
    "_digits": "༠༡༢༣༤༥༦༧༨༩",
    "_type": "numeric"
  },
  "tirh": {

```

```
{
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
},
{
    "vaii": {
        "_digits": "᠑|᠒᠔᠖᠐᠓᠕ᠨᠢᠯᠤᠰ",
        "_type": "numeric"
    },
    "wara": {
        "_digits": "□□□□□□□□□□",
        "_type": "numeric"
    }
}
}
```

NUMBERINGSYSTEMS.JSX

```
{
    "supplemental";
    {
        "version";
        {
            "_number";
            "$Revision: 12732 $",
            "unicodeVersion";
            "9.0.0",
            "_cldrVersion";
            "31";
        }
        "numberingSystems";
        {
            "adlm";
            {
                "_digits";
                "ᐃᐅᐆᐇᐈᐉᑦᑖᑗᑘᑙ",
                "_type";
                "numeric";
            }
            "ahom";
            {
                "_digits";
                "ᩌᩍᩎᩏᩐᩑᩒᩓᩔᩕᩖᩗᩘᩙᩚᩛᩜᩝᩞ᩟᩠ᩡᩢᩣᩤᩥᩦᩧᩨᩩᩪᩫᩬᩭᩮᩯᩰᩱᩲᩳᩴ᩵᩶᩷᩸᩹᩺᩻᩼᩽᩾᩿",
                "_type";
                "numeric";
            }
            "arab";
            {
                "_digits";
                "٠١٢٣٤٥٦٧٨٩",
                "_type";
                "numeric";
            }
            "arabext";
            {
                "_digits";
                "٠١٢٣٤٥٦٧٨٩",
```



```

        "_type";
        "numeric";
    }
    "armn";
    {
        "_rules";
        "armenian-upper",
        "_type";
        "algorithmic";
    }
    "armnlow";
    {
        "_rules";
        "armenian-lower",
        "_type";
        "algorithmic";
    }
    "bali";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "beng";
    {
        "_digits";
        "০১২৩৪৫৬৭৮৯",
        "_type";
        "numeric";
    }
    "bhks";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "brah";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "cakm";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "cham";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",

```

```
        "_type";
        "numeric";
    }
    "cyrl";
    {
        "_rules";
        "cyrillic-lower",
        "_type";
        "algorithmic";
    }
    "deva";
    {
        "_digits";
        "०१२३४५६७८९",
        "_type";
        "numeric";
    }
    "ethi";
    {
        "_rules";
        "ethiopic",
        "_type";
        "algorithmic";
    }
    "fullwide";
    {
        "_digits";
        "0 1 2 3 4 5 6 7 8 9",
        "_type";
        "numeric";
    }
    "geor";
    {
        "_rules";
        "georgian",
        "_type";
        "algorithmic";
    }
    "grek";
    {
        "_rules";
        "greek-upper",
        "_type";
        "algorithmic";
    }
    "greklow";
    {
        "_rules";
        "greek-lower",
        "_type";
        "algorithmic";
    }
    "gujr";
    {
        "_digits";
        "૦૧૨૩૪૫૬૭૮૯",
```

```

        "_type";
        "numeric";
    }
    "guru";
    {
        "_digits";
        "୦୧୨୩୪୫୬୭୮୯",
        "_type";
        "numeric";
    }
    "hanidays";
    {
        "_rules";
        "zh/SpelloutRules/spellout-numbering-days",
        "_type";
        "algorithmic";
    }
    "hanidec";
    {
        "_digits";
        "〇一二三四五六七八九",
        "_type";
        "numeric";
    }
    "hans";
    {
        "_rules";
        "zh/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "hansfin";
    {
        "_rules";
        "zh/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "hant";
    {
        "_rules";
        "zh_Hant/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "hantfin";
    {
        "_rules";
        "zh_Hant/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "hebr";
    {
        "_rules";
        "hebrew",

```

```

        "_type";
        "algorithmic";
    }
    "hmng";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "java";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "jpan";
    {
        "_rules";
        "ja/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "jpanfin";
    {
        "_rules";
        "ja/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "kali";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "khdr";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "knda";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "lana";
    {
        "_digits";

```

```

        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "lanatham";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "laoo";
    {
        "_digits";
        "໐໑໒໓໔໕໖໗໘໑",
        "_type";
        "numeric";
    }
    "latn";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "lepc";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "limb";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "mathbold";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathdbl";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathmono";
    {
        "_digits";

```

```

        "0123456789",
        "_type";
        "numeric";
    }
    "mathsanb";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathsans";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mlym";
    {
        "_digits";
        "ഐറുനൂറു",
        "_type";
        "numeric";
    }
    "modi";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "mong";
    {
        "_digits";
        "᠎ᠠᠨᠣᠭ",
        "_type";
        "numeric";
    }
    "mroo";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "mtei";
    {
        "_digits";
        "၆၆၆၆၆၆၆၆",
        "_type";
        "numeric";
    }
    "mymr";
    {
        "_digits";

```

```

        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "mymrshan";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "mymrtlng";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "newa";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "nkoo";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "olck";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "orya";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "osma";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "roman";

```

```

    {
      "_rules";
      "roman-upper",
      "_type";
      "algorithmic";
    }
    "romanlow";
    {
      "_rules";
      "roman-lower",
      "_type";
      "algorithmic";
    }
    "saur";
    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "shrd";
    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "sind";
    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "sinh";
    {
      "_digits";
      "𑆑𑆒𑆓𑆔𑆕𑆖𑆗𑆘",
      "_type";
      "numeric";
    }
    "sora";
    {
      "_digits";
      "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉",
      "_type";
      "numeric";
    }
    "sund";
    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "takr";

```



```

    {
      "_digits";
      "□□□□□□□□□□",
      "_type";
      "numeric";
    }
    "talu";
    {
      "_digits";
      "ᱠᱟᱰᱟᱨᱢᱟᱝᱞᱟᱹ",
      "_type";
      "numeric";
    }
    "taml";
    {
      "_rules";
      "tamil",
      "_type";
      "algorithmic";
    }
    "tamldec";
    {
      "_digits";
      "௦௧௨௩௪௫௬௭௮௯",
      "_type";
      "numeric";
    }
    "telu";
    {
      "_digits";
      "౦౧౨౩౪౫౬౭౮౯",
      "_type";
      "numeric";
    }
    "thai";
    {
      "_digits";
      "๐๑๒๓๔๕๖๗๘๙",
      "_type";
      "numeric";
    }
    "tibt";
    {
      "_digits";
      "༠༡༢༣༤༥༦༧༨༩",
      "_type";
      "numeric";
    }
    "tirh";
    {
      "_digits";
      "ᱠᱟᱰᱟᱨᱢᱟᱝᱞᱟᱹ",
      "_type";
      "numeric";
    }
    "vaih";

```

```

        {
            "_digits":
            "᠑᠒᠓᠔᠕᠖᠗᠘᠐᠐᠑",
            "_type":
            "numeric";
        }
        "wara";
        {
            "_digits":
            "᠑᠒᠓᠔᠕᠖᠗᠘᠐᠐᠑",
            "_type":
            "numeric";
        }
    }
}

```

NUMBERS.JSON

```

{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31"
        },
        "language": "de"
      },
      "numbers": {
        "defaultNumberingSystem": "latn",
        "otherNumberingSystems": {
          "native": "latn"
        },
        "minimumGroupingDigits": "1",
        "symbols-numberSystem-latn": {
          "decimal": ",",
          "group": ".",
          "list": ";",
          "percentSign": "%",
          "plusSign": "+",
          "minusSign": "-",
          "exponential": "E",
          "superscriptingExponent": "·",
          "perMille": "‰",
          "infinity": "∞",
          "nan": "NaN",
          "timeSeparator": ":"
        },
        "decimalFormats-numberSystem-latn": {
          "standard": "#,##0.###",
          "long": {
            "decimalFormat": {
              "1000-count-one": "0 Tausend",
              "1000-count-other": "0 Tausend",
              "10000-count-one": "00 Tausend",

```

```

        "10000-count-other": "00 Tausend",
        "100000-count-one": "000 Tausend",
        "100000-count-other": "000 Tausend",
        "1000000-count-one": "0 Million",
        "1000000-count-other": "0 Millionen",
        "10000000-count-one": "00 Millionen",
        "10000000-count-other": "00 Millionen",
        "100000000-count-one": "000 Millionen",
        "100000000-count-other": "000 Millionen",
        "1000000000-count-one": "0 Milliarden",
        "1000000000-count-other": "0 Milliarden",
        "10000000000-count-one": "00 Milliarden",
        "10000000000-count-other": "00 Milliarden",
        "100000000000-count-one": "000 Milliarden",
        "100000000000-count-other": "000 Milliarden",
        "1000000000000-count-one": "0 Billion",
        "1000000000000-count-other": "0 Billionen",
        "10000000000000-count-one": "00 Billionen",
        "10000000000000-count-other": "00 Billionen",
        "100000000000000-count-one": "000 Billionen",
        "100000000000000-count-other": "000 Billionen"
    }
},
"short": {
    "decimalFormat": {
        "1000-count-one": "0",
        "1000-count-other": "0",
        "10000-count-one": "0",
        "10000-count-other": "0",
        "100000-count-one": "0",
        "100000-count-other": "0",
        "1000000-count-one": "0 Mio'.'",
        "1000000-count-other": "0 Mio'.'",
        "10000000-count-one": "00 Mio'.'",
        "10000000-count-other": "00 Mio'.'",
        "100000000-count-one": "000 Mio'.'",
        "100000000-count-other": "000 Mio'.'",
        "1000000000-count-one": "0 Mrd'.'",
        "1000000000-count-other": "0 Mrd'.'",
        "10000000000-count-one": "00 Mrd'.'",
        "10000000000-count-other": "00 Mrd'.'",
        "100000000000-count-one": "000 Mrd'.'",
        "100000000000-count-other": "000 Mrd'.'",
        "1000000000000-count-one": "0 Bio'.'",
        "1000000000000-count-other": "0 Bio'.'",
        "10000000000000-count-one": "00 Bio'.'",
        "10000000000000-count-other": "00 Bio'.'",
        "100000000000000-count-one": "000 Bio'.'",
        "100000000000000-count-other": "000 Bio'.'"
    }
},
},
"scientificFormats-numberSystem-latn": {
    "standard": "#E0"
},
"percentFormats-numberSystem-latn": {
    "standard": "#,##0 %"
}

```

```

    },
    "currencyFormats-numberSystem-latn": {
      "currencySpacing": {
        "beforeCurrency": {
          "currencyMatch": "[:^S:]",
          "surroundingMatch": "[:digit:]",
          "insertBetween": " "
        },
        "afterCurrency": {
          "currencyMatch": "[:^S:]",
          "surroundingMatch": "[:digit:]",
          "insertBetween": " "
        }
      },
      "standard": "#,##0.00 ¤",
      "accounting": "#,##0.00 ¤",
      "short": {
        "standard": {
          "1000-count-one": "0 Tsd'.' ¤",
          "1000-count-other": "0 Tsd'.' ¤",
          "10000-count-one": "00 Tsd'.' ¤",
          "10000-count-other": "00 Tsd'.' ¤",
          "100000-count-one": "000 Tsd'.' ¤",
          "100000-count-other": "000 Tsd'.' ¤",
          "1000000-count-one": "0 Mio'.' ¤",
          "1000000-count-other": "0 Mio'.' ¤",
          "10000000-count-one": "00 Mio'.' ¤",
          "10000000-count-other": "00 Mio'.' ¤",
          "100000000-count-one": "000 Mio'.' ¤",
          "100000000-count-other": "000 Mio'.' ¤",
          "1000000000-count-one": "0 Mrd'.' ¤",
          "1000000000-count-other": "0 Mrd'.' ¤",
          "10000000000-count-one": "00 Mrd'.' ¤",
          "10000000000-count-other": "00 Mrd'.' ¤",
          "100000000000-count-one": "000 Mrd'.' ¤",
          "100000000000-count-other": "000 Mrd'.' ¤",
          "1000000000000-count-one": "0 Bio'.' ¤",
          "1000000000000-count-other": "0 Bio'.' ¤",
          "10000000000000-count-one": "00 Bio'.' ¤",
          "10000000000000-count-other": "00 Bio'.' ¤",
          "100000000000000-count-one": "000 Bio'.' ¤",
          "100000000000000-count-other": "000 Bio'.' ¤"
        }
      },
      "unitPattern-count-one": "{0} {1}",
      "unitPattern-count-other": "{0} {1}"
    },
    "miscPatterns-numberSystem-latn": {
      "atLeast": "{0}+",
      "range": "{0}-{1}"
    },
    "minimalPairs": {
      "pluralMinimalPairs": "{0} Tag",
      "pluralMinimalPairs": "{0} Tage",
      "other": "{0}. Abzweigung nach rechts nehmen"
    }
  }
}

```

```
}  
}  
}
```

NUMBERS.JSX

```
{  
  "main";  
  {  
    "de";  
    {  
      "identity";  
      {  
        "version";  
        {  
          "_number";  
          "$Revision: 13259 $",  
          "_cldrVersion";  
          "31";  
        }  
        "language";  
        "de";  
      }  
    }  
    "numbers";  
    {  
      "defaultNumberingSystem";  
      "latn",  
      "otherNumberingSystems";  
      {  
        "native";  
        "latn";  
      }  
      "minimumGroupingDigits";  
      "1",  
      "symbols-numberSystem-latn";  
      {  
        "decimal";  
        ",",  
        "group";  
        ".",  
        "list";  
        ";",  
        "percentSign";  
        "%",  
        "plusSign";  
        "+",  
        "minusSign";  
        "-",  
        "exponential";  
        "E",  
        "superscriptingExponent";  
        ".",  
        "perMille";  
        "‰",  
        "infinity";  
        "∞",  
      }  
    }  
  }  
}
```

```

        "nan";
        "NaN",
        "timeSeparator";
        ":";
    }
    "decimalFormats-numberSystem-latn";
    {
        "standard";
        "#,##0.###",
        "long";
        {
            "decimalFormat";
            {
                "1000-count-one";
                "0 Tausend",
                "1000-count-other";
                "0 Tausend",
                "10000-count-one";
                "00 Tausend",
                "10000-count-other";
                "00 Tausend",
                "100000-count-one";
                "000 Tausend",
                "100000-count-other";
                "000 Tausend",
                "1000000-count-one";
                "0 Million",
                "1000000-count-other";
                "0 Millionen",
                "10000000-count-one";
                "00 Millionen",
                "10000000-count-other";
                "00 Millionen",
                "100000000-count-one";
                "000 Millionen",
                "100000000-count-other";
                "000 Millionen",
                "1000000000-count-one";
                "0 Milliarde",
                "1000000000-count-other";
                "0 Milliarden",
                "10000000000-count-one";
                "00 Milliarden",
                "10000000000-count-other";
                "00 Milliarden",
                "100000000000-count-one";
                "000 Milliarden",
                "100000000000-count-other";
                "000 Milliarden",
                "1000000000000-count-one";
                "0 Billion",
                "1000000000000-count-other";
                "0 Billionen",
                "10000000000000-count-one";
                "00 Billionen",
                "10000000000000-count-other";
                "00 Billionen",
            }
        }
    }

```

```

        "1000000000000000-count-one";
        "000 Billionen",
        "1000000000000000-count-other";
        "000 Billionen";
    }
}
"short";
{
    "decimalFormat";
    {
        "1000-count-one";
        "0",
        "1000-count-other";
        "0",
        "10000-count-one";
        "0",
        "10000-count-other";
        "0",
        "100000-count-one";
        "0",
        "100000-count-other";
        "0",
        "1000000-count-one";
        "0",
        "1000000-count-other";
        "0",
        "1000000-count-one";
        "0 Mio'. '",
        "1000000-count-other";
        "0 Mio'. '",
        "10000000-count-one";
        "00 Mio'. '",
        "10000000-count-other";
        "00 Mio'. '",
        "100000000-count-one";
        "000 Mio'. '",
        "100000000-count-other";
        "000 Mio'. '",
        "1000000000-count-one";
        "0 Mrd'. '",
        "1000000000-count-other";
        "0 Mrd'. '",
        "10000000000-count-one";
        "00 Mrd'. '",
        "10000000000-count-other";
        "00 Mrd'. '",
        "100000000000-count-one";
        "000 Mrd'. '",
        "100000000000-count-other";
        "000 Mrd'. '",
        "1000000000000-count-one";
        "0 Bio'. '",
        "1000000000000-count-other";
        "0 Bio'. '",
        "10000000000000-count-one";
        "00 Bio'. '",
        "10000000000000-count-other";
        "00 Bio'. '",
        "100000000000000-count-one";
        "000 Bio'. '",
        "100000000000000-count-other";
    }
}

```

```

        "000 Bio'.'";
    }
}
"scientificFormats-numberSystem-latn";
{
    "standard";
    "#E0";
}
"percentFormats-numberSystem-latn";
{
    "standard";
    "#,##0 %";
}
"currencyFormats-numberSystem-latn";
{
    "currencySpacing";
    {
        "beforeCurrency";
        {
            "currencyMatch";
            "[:^S:]",
            "surroundingMatch";
            "[:digit:]",
            "insertBetween";
            " ";
        }
        "afterCurrency";
        {
            "currencyMatch";
            "[:^S:]",
            "surroundingMatch";
            "[:digit:]",
            "insertBetween";
            " ";
        }
    }
}
"standard";
"#,##0.00 ¤",
    "accounting";
"#,##0.00 ¤",
    "short";
{
    "standard";
    {
        "1000-count-one";
        "0 Tsd'.' ¤",
        "1000-count-other";
        "0 Tsd'.' ¤",
        "10000-count-one";
        "00 Tsd'.' ¤",
        "10000-count-other";
        "00 Tsd'.' ¤",
        "100000-count-one";
        "000 Tsd'.' ¤",
        "100000-count-other";
        "000 Tsd'.' ¤",
    }
}

```



```

        "1000000-count-one";
        "0 Mio'.' ¤",
        "1000000-count-other";
        "0 Mio'.' ¤",
        "10000000-count-one";
        "00 Mio'.' ¤",
        "10000000-count-other";
        "00 Mio'.' ¤",
        "100000000-count-one";
        "000 Mio'.' ¤",
        "100000000-count-other";
        "000 Mio'.' ¤",
        "1000000000-count-one";
        "0 Mrd'.' ¤",
        "1000000000-count-other";
        "0 Mrd'.' ¤",
        "10000000000-count-one";
        "00 Mrd'.' ¤",
        "10000000000-count-other";
        "00 Mrd'.' ¤",
        "100000000000-count-one";
        "000 Mrd'.' ¤",
        "100000000000-count-other";
        "000 Mrd'.' ¤",
        "1000000000000-count-one";
        "0 Bio'.' ¤",
        "1000000000000-count-other";
        "0 Bio'.' ¤",
        "10000000000000-count-one";
        "00 Bio'.' ¤",
        "10000000000000-count-other";
        "00 Bio'.' ¤",
        "100000000000000-count-one";
        "000 Bio'.' ¤",
        "100000000000000-count-other";
        "000 Bio'.' ¤";
    }
}
"unitPattern-count-one";
"{0} {1}",
    "unitPattern-count-other";
"{0} {1}";
}
"miscPatterns-numberSystem-latn";
{
    "atLeast";
    "{0}+",
        "range";
    "{0}-{1}";
}
"minimalPairs";
{
    "pluralMinimalPairs";
    "{0} Tag",
        "pluralMinimalPairs";
    "{0} Tage",
        "other";
}

```

```

        "{0}. Abzweigung nach rechts nehmen";
    }
    }
}
}
}

```

TIMEZONENAMES.JSON

```

{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 12879 $",
          "_cldrVersion": "30.0.3"
        },
        "language": "de"
      },
      "dates": {
        "timeZoneNames": {
          "hourFormat": "+HH:mm;-HH:mm",
          "gmtFormat": "GMT{0}",
          "gmtZeroFormat": "GMT",
          "regionFormat": "{0} Zeit",
          "regionFormat-type-daylight": "{0} Sommerzeit",
          "regionFormat-type-standard": "{0} Normalzeit",
          "fallbackFormat": "{1} ({0})",
          "zone": {
            "America": {
              "Adak": {
                "exemplarCity": "Adak"
              },
              "Anchorage": {
                "exemplarCity": "Anchorage"
              },
              "Anguilla": {
                "exemplarCity": "Anguilla"
              },
              "Antigua": {
                "exemplarCity": "Antigua"
              },
              "Araguaina": {
                "exemplarCity": "Araguaina"
              },
              "Argentina": {
                "Rio_Gallegos": {
                  "exemplarCity": "Rio Gallegos"
                },
                "San_Juan": {
                  "exemplarCity": "San Juan"
                },
                "Ushuaia": {
                  "exemplarCity": "Ushuaia"
                },
                "La_Rioja": {

```

```
        "exemplarCity": "La Rioja"
      },
      "San_Luis": {
        "exemplarCity": "San Luis"
      },
      "Salta": {
        "exemplarCity": "Salta"
      },
      "Tucuman": {
        "exemplarCity": "Tucuman"
      }
    },
    "Aruba": {
      "exemplarCity": "Aruba"
    },
    "Asuncion": {
      "exemplarCity": "Asunción"
    },
    "Bahia": {
      "exemplarCity": "Bahia"
    },
    "Bahia_Banderas": {
      "exemplarCity": "Bahia Banderas"
    },
    "Barbados": {
      "exemplarCity": "Barbados"
    },
    "Belem": {
      "exemplarCity": "Belem"
    },
    "Belize": {
      "exemplarCity": "Belize"
    },
    "Blanc-Sablon": {
      "exemplarCity": "Blanc-Sablon"
    },
    "Boa_Vista": {
      "exemplarCity": "Boa Vista"
    },
    "Bogota": {
      "exemplarCity": "Bogotá"
    },
    "Boise": {
      "exemplarCity": "Boise"
    },
    "Buenos_Aires": {
      "exemplarCity": "Buenos Aires"
    },
    "Cambridge_Bay": {
      "exemplarCity": "Cambridge Bay"
    },
    "Campo_Grande": {
      "exemplarCity": "Campo Grande"
    },
    "Cancun": {
      "exemplarCity": "Cancún"
    },
  },
```

```
"Caracas": {
  "exemplarCity": "Caracas"
},
"Catamarca": {
  "exemplarCity": "Catamarca"
},
"Cayenne": {
  "exemplarCity": "Cayenne"
},
"Cayman": {
  "exemplarCity": "Kaimaninseln"
},
"Chicago": {
  "exemplarCity": "Chicago"
},
"Chihuahua": {
  "exemplarCity": "Chihuahua"
},
"Coral_Harbour": {
  "exemplarCity": "Atikokan"
},
"Cordoba": {
  "exemplarCity": "Córdoba"
},
"Costa_Rica": {
  "exemplarCity": "Costa Rica"
},
"Creston": {
  "exemplarCity": "Creston"
},
"Cuiaba": {
  "exemplarCity": "Cuiaba"
},
"Curacao": {
  "exemplarCity": "Curaçao"
},
"Danmarkshavn": {
  "exemplarCity": "Danmarkshavn"
},
"Dawson": {
  "exemplarCity": "Dawson"
},
"Dawson_Creek": {
  "exemplarCity": "Dawson Creek"
},
"Denver": {
  "exemplarCity": "Denver"
},
"Detroit": {
  "exemplarCity": "Detroit"
},
"Dominica": {
  "exemplarCity": "Dominica"
},
"Edmonton": {
  "exemplarCity": "Edmonton"
},
}
```

```
"Eirunepe": {
  "exemplarCity": "Eirunepe"
},
"El_Salvador": {
  "exemplarCity": "El Salvador"
},
"Fort_Nelson": {
  "exemplarCity": "Fort Nelson"
},
"Fortaleza": {
  "exemplarCity": "Fortaleza"
},
"Glace_Bay": {
  "exemplarCity": "Glace Bay"
},
"Godthab": {
  "exemplarCity": "Nuuk"
},
"Goose_Bay": {
  "exemplarCity": "Goose Bay"
},
"Grand_Turk": {
  "exemplarCity": "Grand Turk"
},
"Grenada": {
  "exemplarCity": "Grenada"
},
"Guadeloupe": {
  "exemplarCity": "Guadeloupe"
},
"Guatemala": {
  "exemplarCity": "Guatemala"
},
"Guayaquil": {
  "exemplarCity": "Guayaquil"
},
"Guyana": {
  "exemplarCity": "Guyana"
},
"Halifax": {
  "exemplarCity": "Halifax"
},
"Havana": {
  "exemplarCity": "Havanna"
},
"Hermosillo": {
  "exemplarCity": "Hermosillo"
},
"Indiana": {
  "Vincennes": {
    "exemplarCity": "Vincennes, Indiana"
  },
  "Petersburg": {
    "exemplarCity": "Petersburg, Indiana"
  },
  "Tell_City": {
    "exemplarCity": "Tell City, Indiana"
  }
}
```

```
    },  
    "Knox": {  
      "exemplarCity": "Knox, Indiana"  
    },  
    "Winamac": {  
      "exemplarCity": "Winamac, Indiana"  
    },  
    "Marengo": {  
      "exemplarCity": "Marengo, Indiana"  
    },  
    "Vevay": {  
      "exemplarCity": "Vevay, Indiana"  
    }  
  },  
  "Indianapolis": {  
    "exemplarCity": "Indianapolis"  
  },  
  "Inuvik": {  
    "exemplarCity": "Inuvik"  
  },  
  "Iqaluit": {  
    "exemplarCity": "Iqaluit"  
  },  
  "Jamaica": {  
    "exemplarCity": "Jamaika"  
  },  
  "Jujuy": {  
    "exemplarCity": "Jujuy"  
  },  
  "Juneau": {  
    "exemplarCity": "Juneau"  
  },  
  "Kentucky": {  
    "Monticello": {  
      "exemplarCity": "Monticello, Kentucky"  
    }  
  },  
  "Kralendijk": {  
    "exemplarCity": "Kralendijk"  
  },  
  "La_Paz": {  
    "exemplarCity": "La Paz"  
  },  
  "Lima": {  
    "exemplarCity": "Lima"  
  },  
  "Los_Angeles": {  
    "exemplarCity": "Los Angeles"  
  },  
  "Louisville": {  
    "exemplarCity": "Louisville"  
  },  
  "Lower_Princes": {  
    "exemplarCity": "Lower Prince's Quarter"  
  },  
  "Maceio": {  
    "exemplarCity": "Maceio"
```

```
    },  
    "Managua": {  
      "exemplarCity": "Managua"  
    },  
    "Manaus": {  
      "exemplarCity": "Manaus"  
    },  
    "Marigot": {  
      "exemplarCity": "Marigot"  
    },  
    "Martinique": {  
      "exemplarCity": "Martinique"  
    },  
    "Matamoros": {  
      "exemplarCity": "Matamoros"  
    },  
    "Mazatlan": {  
      "exemplarCity": "Mazatlan"  
    },  
    "Mendoza": {  
      "exemplarCity": "Mendoza"  
    },  
    "Menominee": {  
      "exemplarCity": "Menominee"  
    },  
    "Merida": {  
      "exemplarCity": "Merida"  
    },  
    "Metlakatla": {  
      "exemplarCity": "Metlakatla"  
    },  
    "Mexico_City": {  
      "exemplarCity": "Mexiko-Stadt"  
    },  
    "Miquelon": {  
      "exemplarCity": "Miquelon"  
    },  
    "Moncton": {  
      "exemplarCity": "Moncton"  
    },  
    "Monterrey": {  
      "exemplarCity": "Monterrey"  
    },  
    "Montevideo": {  
      "exemplarCity": "Montevideo"  
    },  
    "Montserrat": {  
      "exemplarCity": "Montserrat"  
    },  
    "Nassau": {  
      "exemplarCity": "Nassau"  
    },  
    "New_York": {  
      "exemplarCity": "New York"  
    },  
    "Nipigon": {  
      "exemplarCity": "Nipigon"
```

```
    },
    "Nome": {
      "exemplarCity": "Nome"
    },
    "Noronha": {
      "exemplarCity": "Noronha"
    },
    "North_Dakota": {
      "Beulah": {
        "exemplarCity": "Beulah, North Dakota"
      },
      "New_Salem": {
        "exemplarCity": "New Salem, North Dakota"
      },
      "Center": {
        "exemplarCity": "Center, North Dakota"
      }
    },
    "Ojinaga": {
      "exemplarCity": "Ojinaga"
    },
    "Panama": {
      "exemplarCity": "Panama"
    },
    "Pangnirtung": {
      "exemplarCity": "Pangnirtung"
    },
    "Paramaribo": {
      "exemplarCity": "Paramaribo"
    },
    "Phoenix": {
      "exemplarCity": "Phoenix"
    },
    "Port-au-Prince": {
      "exemplarCity": "Port-au-Prince"
    },
    "Port_of_Spain": {
      "exemplarCity": "Port of Spain"
    },
    "Porto_Velho": {
      "exemplarCity": "Porto Velho"
    },
    "Puerto_Rico": {
      "exemplarCity": "Puerto Rico"
    },
    "Rainy_River": {
      "exemplarCity": "Rainy River"
    },
    "Rankin_Inlet": {
      "exemplarCity": "Rankin Inlet"
    },
    "Recife": {
      "exemplarCity": "Recife"
    },
    "Regina": {
      "exemplarCity": "Regina"
    },
  },
```



```
"Resolute": {
  "exemplarCity": "Resolute"
},
"Rio_Branco": {
  "exemplarCity": "Rio Branco"
},
"Santa_Isabel": {
  "exemplarCity": "Santa Isabel"
},
"Santarem": {
  "exemplarCity": "Santarem"
},
"Santiago": {
  "exemplarCity": "Santiago"
},
"Santo_Domingo": {
  "exemplarCity": "Santo Domingo"
},
"Sao_Paulo": {
  "exemplarCity": "São Paulo"
},
"Scoresbysund": {
  "exemplarCity": "Ittoqqortoormiit"
},
"Sitka": {
  "exemplarCity": "Sitka"
},
"St_Barthelemy": {
  "exemplarCity": "Saint-Barthélemy"
},
"St_Johns": {
  "exemplarCity": "St. John's"
},
"St_Kitts": {
  "exemplarCity": "St. Kitts"
},
"St_Lucia": {
  "exemplarCity": "St. Lucia"
},
"St_Thomas": {
  "exemplarCity": "St. Thomas"
},
"St_Vincent": {
  "exemplarCity": "St. Vincent"
},
"Swift_Current": {
  "exemplarCity": "Swift Current"
},
"Tegucigalpa": {
  "exemplarCity": "Tegucigalpa"
},
"Thule": {
  "exemplarCity": "Thule"
},
"Thunder_Bay": {
  "exemplarCity": "Thunder Bay"
},
```

```
"Tijuana": {
  "exemplarCity": "Tijuana"
},
"Toronto": {
  "exemplarCity": "Toronto"
},
"Tortola": {
  "exemplarCity": "Tortola"
},
"Vancouver": {
  "exemplarCity": "Vancouver"
},
"Whitehorse": {
  "exemplarCity": "Whitehorse"
},
"Winnipeg": {
  "exemplarCity": "Winnipeg"
},
"Yakutat": {
  "exemplarCity": "Yakutat"
},
"Yellowknife": {
  "exemplarCity": "Yellowknife"
}
},
"Atlantic": {
  "Azores": {
    "exemplarCity": "Azoren"
  },
  "Bermuda": {
    "exemplarCity": "Bermudas"
  },
  "Canary": {
    "exemplarCity": "Kanaren"
  },
  "Cape_Verde": {
    "exemplarCity": "Cabo Verde"
  },
  "Faeroe": {
    "exemplarCity": "Färöer"
  },
  "Madeira": {
    "exemplarCity": "Madeira"
  },
  "Reykjavik": {
    "exemplarCity": "Reykjavík"
  },
  "South_Georgia": {
    "exemplarCity": "Südgeorgien"
  },
  "St_Helena": {
    "exemplarCity": "St. Helena"
  },
  "Stanley": {
    "exemplarCity": "Stanley"
  }
},
},
```

```
"Europe": {
  "Amsterdam": {
    "exemplarCity": "Amsterdam"
  },
  "Andorra": {
    "exemplarCity": "Andorra"
  },
  "Astrakhan": {
    "exemplarCity": "Astrachan"
  },
  "Athens": {
    "exemplarCity": "Athen"
  },
  "Belgrade": {
    "exemplarCity": "Belgrad"
  },
  "Berlin": {
    "exemplarCity": "Berlin"
  },
  "Bratislava": {
    "exemplarCity": "Bratislava"
  },
  "Brussels": {
    "exemplarCity": "Brüssel"
  },
  "Bucharest": {
    "exemplarCity": "Bukarest"
  },
  "Budapest": {
    "exemplarCity": "Budapest"
  },
  "Busingen": {
    "exemplarCity": "Büsingen"
  },
  "Chisinau": {
    "exemplarCity": "Kischinau"
  },
  "Copenhagen": {
    "exemplarCity": "Kopenhagen"
  },
  "Dublin": {
    "long": {
      "daylight": "Irische Sommerzeit"
    },
    "exemplarCity": "Dublin"
  },
  "Gibraltar": {
    "exemplarCity": "Gibraltar"
  },
  "Guernsey": {
    "exemplarCity": "Guernsey"
  },
  "Helsinki": {
    "exemplarCity": "Helsinki"
  },
  "Isle_of_Man": {
    "exemplarCity": "Isle of Man"
  }
}
```

```
    },
    "Istanbul": {
      "exemplarCity": "Istanbul"
    },
    "Jersey": {
      "exemplarCity": "Jersey"
    },
    "Kaliningrad": {
      "exemplarCity": "Kaliningrad"
    },
    "Kiev": {
      "exemplarCity": "Kiew"
    },
    "Kirov": {
      "exemplarCity": "Kirow"
    },
    "Lisbon": {
      "exemplarCity": "Lissabon"
    },
    "Ljubljana": {
      "exemplarCity": "Ljubljana"
    },
    "London": {
      "long": {
        "daylight": "Britische Sommerzeit"
      },
      "exemplarCity": "London"
    },
    "Luxembourg": {
      "exemplarCity": "Luxemburg"
    },
    "Madrid": {
      "exemplarCity": "Madrid"
    },
    "Malta": {
      "exemplarCity": "Malta"
    },
    "Mariehamn": {
      "exemplarCity": "Mariehamn"
    },
    "Minsk": {
      "exemplarCity": "Minsk"
    },
    "Monaco": {
      "exemplarCity": "Monaco"
    },
    "Moscow": {
      "exemplarCity": "Moskau"
    },
    "Oslo": {
      "exemplarCity": "Oslo"
    },
    "Paris": {
      "exemplarCity": "Paris"
    },
    "Podgorica": {
      "exemplarCity": "Podgorica"
    }
  },
  "Istanbul": {
    "exemplarCity": "Istanbul"
  },
  "Jersey": {
    "exemplarCity": "Jersey"
  },
  "Kaliningrad": {
    "exemplarCity": "Kaliningrad"
  },
  "Kiev": {
    "exemplarCity": "Kiew"
  },
  "Kirov": {
    "exemplarCity": "Kirow"
  },
  "Lisbon": {
    "exemplarCity": "Lissabon"
  },
  "Ljubljana": {
    "exemplarCity": "Ljubljana"
  },
  "London": {
    "long": {
      "daylight": "Britische Sommerzeit"
    },
    "exemplarCity": "London"
  },
  "Luxembourg": {
    "exemplarCity": "Luxemburg"
  },
  "Madrid": {
    "exemplarCity": "Madrid"
  },
  "Malta": {
    "exemplarCity": "Malta"
  },
  "Mariehamn": {
    "exemplarCity": "Mariehamn"
  },
  "Minsk": {
    "exemplarCity": "Minsk"
  },
  "Monaco": {
    "exemplarCity": "Monaco"
  },
  "Moscow": {
    "exemplarCity": "Moskau"
  },
  "Oslo": {
    "exemplarCity": "Oslo"
  },
  "Paris": {
    "exemplarCity": "Paris"
  },
  "Podgorica": {
    "exemplarCity": "Podgorica"
  }
}
```

```
    },  
    "Prague": {  
      "exemplarCity": "Prag"  
    },  
    "Riga": {  
      "exemplarCity": "Riga"  
    },  
    "Rome": {  
      "exemplarCity": "Rom"  
    },  
    "Samara": {  
      "exemplarCity": "Samara"  
    },  
    "San_Marino": {  
      "exemplarCity": "San Marino"  
    },  
    "Sarajevo": {  
      "exemplarCity": "Sarajevo"  
    },  
    "Simferopol": {  
      "exemplarCity": "Simferopol"  
    },  
    "Skopje": {  
      "exemplarCity": "Skopje"  
    },  
    "Sofia": {  
      "exemplarCity": "Sofia"  
    },  
    "Stockholm": {  
      "exemplarCity": "Stockholm"  
    },  
    "Tallinn": {  
      "exemplarCity": "Tallinn"  
    },  
    "Tirane": {  
      "exemplarCity": "Tirana"  
    },  
    "Ulyanovsk": {  
      "exemplarCity": "Uljanowsk"  
    },  
    "Uzhgorod": {  
      "exemplarCity": "Uschgorod"  
    },  
    "Vaduz": {  
      "exemplarCity": "Vaduz"  
    },  
    "Vatican": {  
      "exemplarCity": "Vatikan"  
    },  
    "Vienna": {  
      "exemplarCity": "Wien"  
    },  
    "Vilnius": {  
      "exemplarCity": "Vilnius"  
    },  
    "Volgograd": {  
      "exemplarCity": "Wolgograd"
```

```
    },  
    "Warsaw": {  
      "exemplarCity": "Warschau"  
    },  
    "Zagreb": {  
      "exemplarCity": "Zagreb"  
    },  
    "Zaporozhye": {  
      "exemplarCity": "Saporischja"  
    },  
    "Zurich": {  
      "exemplarCity": "Zürich"  
    }  
  },  
  "Africa": {  
    "Abidjan": {  
      "exemplarCity": "Abidjan"  
    },  
    "Accra": {  
      "exemplarCity": "Accra"  
    },  
    "Addis_Ababa": {  
      "exemplarCity": "Addis Abeba"  
    },  
    "Algiers": {  
      "exemplarCity": "Algier"  
    },  
    "Asmera": {  
      "exemplarCity": "Asmara"  
    },  
    "Bamako": {  
      "exemplarCity": "Bamako"  
    },  
    "Bangui": {  
      "exemplarCity": "Bangui"  
    },  
    "Banjul": {  
      "exemplarCity": "Banjul"  
    },  
    "Bissau": {  
      "exemplarCity": "Bissau"  
    },  
    "Blantyre": {  
      "exemplarCity": "Blantyre"  
    },  
    "Brazzaville": {  
      "exemplarCity": "Brazzaville"  
    },  
    "Bujumbura": {  
      "exemplarCity": "Bujumbura"  
    },  
    "Cairo": {  
      "exemplarCity": "Kairo"  
    },  
    "Casablanca": {  
      "exemplarCity": "Casablanca"  
    }  
  },  
}
```

```
"Ceuta": {
  "exemplarCity": "Ceuta"
},
"Conakry": {
  "exemplarCity": "Conakry"
},
"Dakar": {
  "exemplarCity": "Dakar"
},
"Dar_es_Salaam": {
  "exemplarCity": "Daressalam"
},
"Djibouti": {
  "exemplarCity": "Dschibuti"
},
"Douala": {
  "exemplarCity": "Douala"
},
"El_Aaiun": {
  "exemplarCity": "El Aaiún"
},
"Freetown": {
  "exemplarCity": "Freetown"
},
"Gaborone": {
  "exemplarCity": "Gaborone"
},
"Harare": {
  "exemplarCity": "Harare"
},
"Johannesburg": {
  "exemplarCity": "Johannesburg"
},
"Juba": {
  "exemplarCity": "Juba"
},
"Kampala": {
  "exemplarCity": "Kampala"
},
"Khartoum": {
  "exemplarCity": "Khartum"
},
"Kigali": {
  "exemplarCity": "Kigali"
},
"Kinshasa": {
  "exemplarCity": "Kinshasa"
},
"Lagos": {
  "exemplarCity": "Lagos"
},
"Libreville": {
  "exemplarCity": "Libreville"
},
"Lome": {
  "exemplarCity": "Lomé"
},
}
```

```
"Luanda": {
  "exemplarCity": "Luanda"
},
"Lubumbashi": {
  "exemplarCity": "Lubumbashi"
},
"Lusaka": {
  "exemplarCity": "Lusaka"
},
"Malabo": {
  "exemplarCity": "Malabo"
},
"Maputo": {
  "exemplarCity": "Maputo"
},
"Maseru": {
  "exemplarCity": "Maseru"
},
"Mbabane": {
  "exemplarCity": "Mbabane"
},
"Mogadishu": {
  "exemplarCity": "Mogadischu"
},
"Monrovia": {
  "exemplarCity": "Monrovia"
},
"Nairobi": {
  "exemplarCity": "Nairobi"
},
"Ndjamena": {
  "exemplarCity": "N'Djamena"
},
"Niamey": {
  "exemplarCity": "Niamey"
},
"Nouakchott": {
  "exemplarCity": "Nouakchott"
},
"Ouagadougou": {
  "exemplarCity": "Ouagadougou"
},
"Porto-Novo": {
  "exemplarCity": "Porto Novo"
},
"Sao_Tome": {
  "exemplarCity": "São Tomé"
},
"Tripoli": {
  "exemplarCity": "Tripolis"
},
"Tunis": {
  "exemplarCity": "Tunis"
},
"Windhoek": {
  "exemplarCity": "Windhoek"
}
```



```
    },  
    "Asia": {  
      "Aden": {  
        "exemplarCity": "Aden"  
      },  
      "Almaty": {  
        "exemplarCity": "Almaty"  
      },  
      "Amman": {  
        "exemplarCity": "Amman"  
      },  
      "Anadyr": {  
        "exemplarCity": "Anadyr"  
      },  
      "Aqtau": {  
        "exemplarCity": "Aqtau"  
      },  
      "Aqtobe": {  
        "exemplarCity": "Aktobe"  
      },  
      "Ashgabat": {  
        "exemplarCity": "Aşgabat"  
      },  
      "Baghdad": {  
        "exemplarCity": "Bagdad"  
      },  
      "Bahrain": {  
        "exemplarCity": "Bahrain"  
      },  
      "Baku": {  
        "exemplarCity": "Baku"  
      },  
      "Bangkok": {  
        "exemplarCity": "Bangkok"  
      },  
      "Barnaul": {  
        "exemplarCity": "Barnaul"  
      },  
      "Beirut": {  
        "exemplarCity": "Beirut"  
      },  
      "Bishkek": {  
        "exemplarCity": "Bischkek"  
      },  
      "Brunei": {  
        "exemplarCity": "Brunei"  
      },  
      "Calcutta": {  
        "exemplarCity": "Kalkutta"  
      },  
      "Chita": {  
        "exemplarCity": "Tschita"  
      },  
      "Choibalsan": {  
        "exemplarCity": "Tschoibalsan"  
      },  
      "Colombo": {
```

```
        "exemplarCity": "Colombo"
      },
      "Damascus": {
        "exemplarCity": "Damaskus"
      },
      "Dhaka": {
        "exemplarCity": "Dhaka"
      },
      "Dili": {
        "exemplarCity": "Dili"
      },
      "Dubai": {
        "exemplarCity": "Dubai"
      },
      "Dushanbe": {
        "exemplarCity": "Duschanbe"
      },
      "Gaza": {
        "exemplarCity": "Gaza"
      },
      "Hebron": {
        "exemplarCity": "Hebron"
      },
      "Hong_Kong": {
        "exemplarCity": "Hongkong"
      },
      "Hovd": {
        "exemplarCity": "Chowd"
      },
      "Irkutsk": {
        "exemplarCity": "Irkutsk"
      },
      "Jakarta": {
        "exemplarCity": "Jakarta"
      },
      "Jayapura": {
        "exemplarCity": "Jayapura"
      },
      "Jerusalem": {
        "exemplarCity": "Jerusalem"
      },
      "Kabul": {
        "exemplarCity": "Kabul"
      },
      "Kamchatka": {
        "exemplarCity": "Kamtschatka"
      },
      "Karachi": {
        "exemplarCity": "Karatschi"
      },
      "Katmandu": {
        "exemplarCity": "Kathmandu"
      },
      "Khandyga": {
        "exemplarCity": "Chandyga"
      },
      "Krasnoyarsk": {
```

```
    "exemplarCity": "Krasnojarsk"
  },
  "Kuala_Lumpur": {
    "exemplarCity": "Kuala Lumpur"
  },
  "Kuching": {
    "exemplarCity": "Kuching"
  },
  "Kuwait": {
    "exemplarCity": "Kuwait"
  },
  "Macau": {
    "exemplarCity": "Macao"
  },
  "Magadan": {
    "exemplarCity": "Magadan"
  },
  "Makassar": {
    "exemplarCity": "Makassar"
  },
  "Manila": {
    "exemplarCity": "Manila"
  },
  "Muscat": {
    "exemplarCity": "Maskat"
  },
  "Nicosia": {
    "exemplarCity": "Nikosia"
  },
  "Novokuznetsk": {
    "exemplarCity": "Nowokuznetsk"
  },
  "Novosibirsk": {
    "exemplarCity": "Nowosibirsk"
  },
  "Omsk": {
    "exemplarCity": "Omsk"
  },
  "Oral": {
    "exemplarCity": "Oral"
  },
  "Phnom_Penh": {
    "exemplarCity": "Phnom Penh"
  },
  "Pontianak": {
    "exemplarCity": "Pontianak"
  },
  "Pyongyang": {
    "exemplarCity": "Pjöngjang"
  },
  "Qatar": {
    "exemplarCity": "Katar"
  },
  "Qyzylorda": {
    "exemplarCity": "Qysylorda"
  },
  "Rangoon": {
```

```
    "exemplarCity": "Rangun"
  },
  "Riyadh": {
    "exemplarCity": "Riad"
  },
  "Saigon": {
    "exemplarCity": "Ho-Chi-Minh-Stadt"
  },
  "Sakhalin": {
    "exemplarCity": "Sachalin"
  },
  "Samarkand": {
    "exemplarCity": "Samarkand"
  },
  "Seoul": {
    "exemplarCity": "Seoul"
  },
  "Shanghai": {
    "exemplarCity": "Shanghai"
  },
  "Singapore": {
    "exemplarCity": "Singapur"
  },
  "Srednekolymensk": {
    "exemplarCity": "Srednekolymensk"
  },
  "Taipei": {
    "exemplarCity": "Taipeh"
  },
  "Tashkent": {
    "exemplarCity": "Taschkent"
  },
  "Tbilisi": {
    "exemplarCity": "Tiflis"
  },
  "Tehran": {
    "exemplarCity": "Teheran"
  },
  "Thimphu": {
    "exemplarCity": "Thimphu"
  },
  "Tokyo": {
    "exemplarCity": "Tokio"
  },
  "Tomsk": {
    "exemplarCity": "Tomsk"
  },
  "Ulaanbaatar": {
    "exemplarCity": "Ulaanbaatar"
  },
  "Urumqi": {
    "exemplarCity": "Ürümqi"
  },
  "Ust-Nera": {
    "exemplarCity": "Ust-Nera"
  },
  "Vientiane": {
```

```
        "exemplarCity": "Vientiane"
      },
      "Vladivostok": {
        "exemplarCity": "Wladiwostok"
      },
      "Yakutsk": {
        "exemplarCity": "Jakutsk"
      },
      "Yekaterinburg": {
        "exemplarCity": "Jekaterinburg"
      },
      "Yerevan": {
        "exemplarCity": "Eriwan"
      }
    },
    "Indian": {
      "Antananarivo": {
        "exemplarCity": "Antananarivo"
      },
      "Chagos": {
        "exemplarCity": "Chagos"
      },
      "Christmas": {
        "exemplarCity": "Weihnachtsinsel"
      },
      "Cocos": {
        "exemplarCity": "Cocos"
      },
      "Comoro": {
        "exemplarCity": "Komoren"
      },
      "Kerguelen": {
        "exemplarCity": "Kerguelen"
      },
      "Mahe": {
        "exemplarCity": "Mahe"
      },
      "Maldives": {
        "exemplarCity": "Malediven"
      },
      "Mauritius": {
        "exemplarCity": "Mauritius"
      },
      "Mayotte": {
        "exemplarCity": "Mayotte"
      },
      "Reunion": {
        "exemplarCity": "Réunion"
      }
    },
    "Australia": {
      "Adelaide": {
        "exemplarCity": "Adelaide"
      },
      "Brisbane": {
        "exemplarCity": "Brisbane"
      }
    }
  },
  "exemplarCity": "Vientiane"
}
```

```
"Broken_Hill": {
  "exemplarCity": "Broken Hill"
},
"Currie": {
  "exemplarCity": "Currie"
},
"Darwin": {
  "exemplarCity": "Darwin"
},
"Eucla": {
  "exemplarCity": "Eucla"
},
"Hobart": {
  "exemplarCity": "Hobart"
},
"Lindeman": {
  "exemplarCity": "Lindeman"
},
"Lord_Howe": {
  "exemplarCity": "Lord Howe"
},
"Melbourne": {
  "exemplarCity": "Melbourne"
},
"Perth": {
  "exemplarCity": "Perth"
},
"Sydney": {
  "exemplarCity": "Sydney"
}
},
"Pacific": {
  "Apia": {
    "exemplarCity": "Apia"
  },
  "Auckland": {
    "exemplarCity": "Auckland"
  },
  "Bougainville": {
    "exemplarCity": "Bougainville"
  },
  "Chatham": {
    "exemplarCity": "Chatham"
  },
  "Easter": {
    "exemplarCity": "Osterinsel"
  },
  "Efate": {
    "exemplarCity": "Efate"
  },
  "Enderbury": {
    "exemplarCity": "Enderbury"
  },
  "Fakaofu": {
    "exemplarCity": "Fakaofu"
  },
  "Fiji": {
```

```
        "exemplarCity": "Fidschi"
      },
      "Funafuti": {
        "exemplarCity": "Funafuti"
      },
      "Galapagos": {
        "exemplarCity": "Galapagos"
      },
      "Gambier": {
        "exemplarCity": "Gambier"
      },
      "Guadalcanal": {
        "exemplarCity": "Guadalcanal"
      },
      "Guam": {
        "exemplarCity": "Guam"
      },
      "Honolulu": {
        "exemplarCity": "Honolulu"
      },
      "Johnston": {
        "exemplarCity": "Johnston"
      },
      "Kiritimati": {
        "exemplarCity": "Kiritimati"
      },
      "Kosrae": {
        "exemplarCity": "Kosrae"
      },
      "Kwajalein": {
        "exemplarCity": "Kwajalein"
      },
      "Majuro": {
        "exemplarCity": "Majuro"
      },
      "Marquesas": {
        "exemplarCity": "Marquesas"
      },
      "Midway": {
        "exemplarCity": "Midway"
      },
      "Nauru": {
        "exemplarCity": "Nauru"
      },
      "Niue": {
        "exemplarCity": "Niue"
      },
      "Norfolk": {
        "exemplarCity": "Norfolk"
      },
      "Noumea": {
        "exemplarCity": "Noumea"
      },
      "Pago_Pago": {
        "exemplarCity": "Pago Pago"
      },
      "Palau": {
```

```
        "exemplarCity": "Palau"
      },
      "Pitcairn": {
        "exemplarCity": "Pitcairn"
      },
      "Ponape": {
        "exemplarCity": "Pohnpei"
      },
      "Port_Moresby": {
        "exemplarCity": "Port Moresby"
      },
      "Rarotonga": {
        "exemplarCity": "Rarotonga"
      },
      "Saipan": {
        "exemplarCity": "Saipan"
      },
      "Tahiti": {
        "exemplarCity": "Tahiti"
      },
      "Tarawa": {
        "exemplarCity": "Tarawa"
      },
      "Tongatapu": {
        "exemplarCity": "Tongatapu"
      },
      "Truk": {
        "exemplarCity": "Chuuk"
      },
      "Wake": {
        "exemplarCity": "Wake"
      },
      "Wallis": {
        "exemplarCity": "Wallis"
      }
    },
    "Arctic": {
      "Longyearbyen": {
        "exemplarCity": "Longyearbyen"
      }
    },
    "Antarctica": {
      "Casey": {
        "exemplarCity": "Casey"
      },
      "Davis": {
        "exemplarCity": "Davis"
      },
      "DumontDUrville": {
        "exemplarCity": "Dumont d'Urville"
      },
      "Macquarie": {
        "exemplarCity": "Macquarie"
      },
      "Mawson": {
        "exemplarCity": "Mawson"
      }
    }
  },
```



```
"McMurdo": {
  "exemplarCity": "McMurdo"
},
"Palmer": {
  "exemplarCity": "Palmer"
},
"Rothera": {
  "exemplarCity": "Rothera"
},
"Syowa": {
  "exemplarCity": "Syowa"
},
"Troll": {
  "exemplarCity": "Troll"
},
"Vostok": {
  "exemplarCity": "Wostok"
}
},
"Etc": {
  "GMT": {
    "exemplarCity": "GMT"
  },
  "GMT1": {
    "exemplarCity": "GMT+1"
  },
  "GMT10": {
    "exemplarCity": "GMT+10"
  },
  "GMT11": {
    "exemplarCity": "GMT+11"
  },
  "GMT12": {
    "exemplarCity": "GMT+12"
  },
  "GMT2": {
    "exemplarCity": "GMT+2"
  },
  "GMT3": {
    "exemplarCity": "GMT+3"
  },
  "GMT4": {
    "exemplarCity": "GMT+4"
  },
  "GMT5": {
    "exemplarCity": "GMT+5"
  },
  "GMT6": {
    "exemplarCity": "GMT+6"
  },
  "GMT7": {
    "exemplarCity": "GMT+7"
  },
  "GMT8": {
    "exemplarCity": "GMT+8"
  },
  "GMT9": {
```

```

        "exemplarCity": "GMT+9"
      },
      "GMT-1": {
        "exemplarCity": "GMT-1"
      },
      "GMT-10": {
        "exemplarCity": "GMT-10"
      },
      "GMT-11": {
        "exemplarCity": "GMT-11"
      },
      "GMT-12": {
        "exemplarCity": "GMT-12"
      },
      "GMT-13": {
        "exemplarCity": "GMT-13"
      },
      "GMT-14": {
        "exemplarCity": "GMT-14"
      },
      "GMT-2": {
        "exemplarCity": "GMT-2"
      },
      "GMT-3": {
        "exemplarCity": "GMT-3"
      },
      "GMT-4": {
        "exemplarCity": "GMT-4"
      },
      "GMT-5": {
        "exemplarCity": "GMT-5"
      },
      "GMT-6": {
        "exemplarCity": "GMT-6"
      },
      "GMT-7": {
        "exemplarCity": "GMT-7"
      },
      "GMT-8": {
        "exemplarCity": "GMT-8"
      },
      "GMT-9": {
        "exemplarCity": "GMT-9"
      },
      "Unknown": {
        "exemplarCity": "Unbekannt"
      }
    },
    "metazone": {
      "Acre": {
        "long": {
          "generic": "Acre-Zeit",
          "standard": "Acre-Normalzeit",
          "daylight": "Acre-Sommerzeit"
        }
      }
    }
  },

```

```
"Afghanistan": {
  "long": {
    "standard": "Afghanistan-Zeit"
  }
},
"Africa_Central": {
  "long": {
    "standard": "Zentralafrikanische Zeit"
  }
},
"Africa_Eastern": {
  "long": {
    "standard": "Ostafrikanische Zeit"
  }
},
"Africa_Southern": {
  "long": {
    "standard": "Südafrikanische Zeit"
  }
},
"Africa_Western": {
  "long": {
    "generic": "Westafrikanische Zeit",
    "standard": "Westafrikanische Normalzeit",
    "daylight": "Westafrikanische Sommerzeit"
  }
},
"Alaska": {
  "long": {
    "generic": "Alaska-Zeit",
    "standard": "Alaska-Normalzeit",
    "daylight": "Alaska-Sommerzeit"
  }
},
"Almaty": {
  "long": {
    "generic": "Almaty-Zeit",
    "standard": "Almaty-Normalzeit",
    "daylight": "Almaty-Sommerzeit"
  }
},
"Amazon": {
  "long": {
    "generic": "Amazonas-Zeit",
    "standard": "Amazonas-Normalzeit",
    "daylight": "Amazonas-Sommerzeit"
  }
},
"America_Central": {
  "long": {
    "generic": "Nordamerikanische Inlandzeit",
    "standard": "Nordamerikanische Inland-Normalzeit",
    "daylight": "Nordamerikanische Inland-Sommerzeit"
  }
},
"America_Eastern": {
  "long": {
```

```

        "generic": "Nordamerikanische Ostküstenzeit",
        "standard": "Nordamerikanische Ostküsten-Normalzeit",
        "daylight": "Nordamerikanische Ostküsten-Sommerzeit"
    }
},
"America_Mountain": {
    "long": {
        "generic": "Rocky-Mountain-Zeit",
        "standard": "Rocky Mountain-Normalzeit",
        "daylight": "Rocky-Mountain-Sommerzeit"
    }
},
"America_Pacific": {
    "long": {
        "generic": "Nordamerikanische Westküstenzeit",
        "standard": "Nordamerikanische Westküsten-Normalzeit",
        "daylight": "Nordamerikanische Westküsten-Sommerzeit"
    }
},
"Anadyr": {
    "long": {
        "generic": "Anadyr Zeit",
        "standard": "Anadyr Normalzeit",
        "daylight": "Anadyr Sommerzeit"
    }
},
"Apia": {
    "long": {
        "generic": "Apia-Zeit",
        "standard": "Apia-Normalzeit",
        "daylight": "Apia-Sommerzeit"
    }
},
"Aqtau": {
    "long": {
        "generic": "Aqtau-Zeit",
        "standard": "Aqtau-Normalzeit",
        "daylight": "Aqtau-Sommerzeit"
    }
},
"Aqtobe": {
    "long": {
        "generic": "Aqtöbe-Zeit",
        "standard": "Aqtöbe-Normalzeit",
        "daylight": "Aqtöbe-Sommerzeit"
    }
},
"Arabian": {
    "long": {
        "generic": "Arabische Zeit",
        "standard": "Arabische Normalzeit",
        "daylight": "Arabische Sommerzeit"
    }
},
"Argentina": {
    "long": {
        "generic": "Argentinische Zeit",

```

```

        "standard": "Argentinische Normalzeit",
        "daylight": "Argentinische Sommerzeit"
    },
    },
    "Argentina_Western": {
        "long": {
            "generic": "Westargentinische Zeit",
            "standard": "Westargentinische Normalzeit",
            "daylight": "Westargentinische Sommerzeit"
        }
    },
    },
    "Armenia": {
        "long": {
            "generic": "Armenische Zeit",
            "standard": "Armenische Normalzeit",
            "daylight": "Armenische Sommerzeit"
        }
    },
    },
    "Atlantic": {
        "long": {
            "generic": "Atlantik-Zeit",
            "standard": "Atlantik-Normalzeit",
            "daylight": "Atlantik-Sommerzeit"
        }
    },
    },
    "Australia_Central": {
        "long": {
            "generic": "Zentralaustralische Zeit",
            "standard": "Zentralaustralische Normalzeit",
            "daylight": "Zentralaustralische Sommerzeit"
        }
    },
    },
    "Australia_CentralWestern": {
        "long": {
            "generic": "Zentral-/Westaustralische Zeit",
            "standard": "Zentral-/Westaustralische Normalzeit",
            "daylight": "Zentral-/Westaustralische Sommerzeit"
        }
    },
    },
    "Australia_Eastern": {
        "long": {
            "generic": "Ostaustralische Zeit",
            "standard": "Ostaustralische Normalzeit",
            "daylight": "Ostaustralische Sommerzeit"
        }
    },
    },
    "Australia_Western": {
        "long": {
            "generic": "Westaustralische Zeit",
            "standard": "Westaustralische Normalzeit",
            "daylight": "Westaustralische Sommerzeit"
        }
    },
    },
    "Azerbaijan": {
        "long": {
            "generic": "Aserbaidshanische Zeit",
            "standard": "Aserbeidschanische Normalzeit",

```

```

        "daylight": "Aserbaidtschanische Sommerzeit"
    },
    "Azores": {
        "long": {
            "generic": "Azoren-Zeit",
            "standard": "Azoren-Normalzeit",
            "daylight": "Azoren-Sommerzeit"
        }
    },
    "Bangladesh": {
        "long": {
            "generic": "Bangladesch-Zeit",
            "standard": "Bangladesch-Normalzeit",
            "daylight": "Bangladesch-Sommerzeit"
        }
    },
    "Bhutan": {
        "long": {
            "standard": "Bhutan-Zeit"
        }
    },
    "Bolivia": {
        "long": {
            "standard": "Bolivianische Zeit"
        }
    },
    "Brasilia": {
        "long": {
            "generic": "Brasília-Zeit",
            "standard": "Brasília-Normalzeit",
            "daylight": "Brasília-Sommerzeit"
        }
    },
    "Brunei": {
        "long": {
            "standard": "Brunei-Zeit"
        }
    },
    "Cape_Verde": {
        "long": {
            "generic": "Cabo-Verde-Zeit",
            "standard": "Cabo-Verde-Normalzeit",
            "daylight": "Cabo-Verde-Sommerzeit"
        }
    },
    "Casey": {
        "long": {
            "standard": "Casey-Zeit"
        }
    },
    "Chamorro": {
        "long": {
            "standard": "Chamorro-Zeit"
        }
    },
    "Chatham": {

```

```
    "long": {
      "generic": "Chatham-Zeit",
      "standard": "Chatham-Normalzeit",
      "daylight": "Chatham-Sommerzeit"
    }
  },
  "Chile": {
    "long": {
      "generic": "Chilenische Zeit",
      "standard": "Chilenische Normalzeit",
      "daylight": "Chilenische Sommerzeit"
    }
  },
  "China": {
    "long": {
      "generic": "Chinesische Zeit",
      "standard": "Chinesische Normalzeit",
      "daylight": "Chinesische Sommerzeit"
    }
  },
  "Choibalsan": {
    "long": {
      "generic": "Tschoibalsan-Zeit",
      "standard": "Tschoibalsan-Normalzeit",
      "daylight": "Tschoibalsan-Sommerzeit"
    }
  },
  "Christmas": {
    "long": {
      "standard": "Weihnachtsinsel-Zeit"
    }
  },
  "Cocos": {
    "long": {
      "standard": "Kokosinseln-Zeit"
    }
  },
  "Colombia": {
    "long": {
      "generic": "Kolumbianische Zeit",
      "standard": "Kolumbianische Normalzeit",
      "daylight": "Kolumbianische Sommerzeit"
    }
  },
  "Cook": {
    "long": {
      "generic": "Cookinseln-Zeit",
      "standard": "Cookinseln-Normalzeit",
      "daylight": "Cookinseln-Sommerzeit"
    }
  },
  "Cuba": {
    "long": {
      "generic": "Kubanische Zeit",
      "standard": "Kubanische Normalzeit",
      "daylight": "Kubanische Sommerzeit"
    }
  }
}
```

```
    },
    "Davis": {
      "long": {
        "standard": "Davis-Zeit"
      }
    },
    "DumontDUrville": {
      "long": {
        "standard": "Dumont-d'Urville-Zeit"
      }
    },
    "East_Timor": {
      "long": {
        "standard": "Osttimor-Zeit"
      }
    },
    "Easter": {
      "long": {
        "generic": "Osterinsel-Zeit",
        "standard": "Osterinsel-Normalzeit",
        "daylight": "Osterinsel-Sommerzeit"
      }
    },
    "Ecuador": {
      "long": {
        "standard": "Ecuadorianische Zeit"
      }
    },
    "Europe_Central": {
      "long": {
        "generic": "Mitteleuropäische Zeit",
        "standard": "Mitteleuropäische Normalzeit",
        "daylight": "Mitteleuropäische Sommerzeit"
      },
      "short": {
        "generic": "MEZ",
        "standard": "MEZ",
        "daylight": "MESZ"
      }
    },
    "Europe_Eastern": {
      "long": {
        "generic": "Osteuropäische Zeit",
        "standard": "Osteuropäische Normalzeit",
        "daylight": "Osteuropäische Sommerzeit"
      },
      "short": {
        "generic": "OEZ",
        "standard": "OEZ",
        "daylight": "OESZ"
      }
    },
    "Europe_Further_Eastern": {
      "long": {
        "standard": "Kaliningrader Zeit"
      }
    }
  },
}
```



```
"Europe_Western": {
  "long": {
    "generic": "Westeuropäische Zeit",
    "standard": "Westeuropäische Normalzeit",
    "daylight": "Westeuropäische Sommerzeit"
  },
  "short": {
    "generic": "WEZ",
    "standard": "WEZ",
    "daylight": "WESZ"
  }
},
"Falkland": {
  "long": {
    "generic": "Falklandinseln-Zeit",
    "standard": "Falklandinseln-Normalzeit",
    "daylight": "Falklandinseln-Sommerzeit"
  }
},
"Fiji": {
  "long": {
    "generic": "Fidschi-Zeit",
    "standard": "Fidschi-Normalzeit",
    "daylight": "Fidschi-Sommerzeit"
  }
},
"French_Guiana": {
  "long": {
    "standard": "Französisch-Guayana-Zeit"
  }
},
"French_Southern": {
  "long": {
    "standard": "Französische Süd- und Antarktisgebiete-Zeit"
  }
},
"Galapagos": {
  "long": {
    "standard": "Galapagos-Zeit"
  }
},
"Gambier": {
  "long": {
    "standard": "Gambier-Zeit"
  }
},
"Georgia": {
  "long": {
    "generic": "Georgische Zeit",
    "standard": "Georgische Normalzeit",
    "daylight": "Georgische Sommerzeit"
  }
},
"Gilbert_Islands": {
  "long": {
    "standard": "Gilbert-Inseln-Zeit"
  }
}
```

```

    },
    "GMT": {
      "long": {
        "standard": "Mittlere Greenwich-Zeit"
      }
    },
    "Greenland_Eastern": {
      "long": {
        "generic": "Ostgrönland-Zeit",
        "standard": "Ostgrönland-Normalzeit",
        "daylight": "Ostgrönland-Sommerzeit"
      }
    },
    "Greenland_Western": {
      "long": {
        "generic": "Westgrönland-Zeit",
        "standard": "Westgrönland-Normalzeit",
        "daylight": "Westgrönland-Sommerzeit"
      }
    },
    "Guam": {
      "long": {
        "standard": "Guam-Zeit"
      }
    },
    "Gulf": {
      "long": {
        "standard": "Golf-Zeit"
      }
    },
    "Guyana": {
      "long": {
        "standard": "Guyana-Zeit"
      }
    },
    "Hawaii_Aleutian": {
      "long": {
        "generic": "Hawaii-Aleuten-Zeit",
        "standard": "Hawaii-Aleuten-Normalzeit",
        "daylight": "Hawaii-Aleuten-Sommerzeit"
      }
    },
    "Hong_Kong": {
      "long": {
        "generic": "Hongkong-Zeit",
        "standard": "Hongkong-Normalzeit",
        "daylight": "Hongkong-Sommerzeit"
      }
    },
    "Hovd": {
      "long": {
        "generic": "Chowd-Zeit",
        "standard": "Chowd-Normalzeit",
        "daylight": "Chowd-Sommerzeit"
      }
    },
    "India": {

```

```

        "long": {
            "standard": "Indische Zeit"
        }
    },
    "Indian_Ocean": {
        "long": {
            "standard": "Indischer Ozean-Zeit"
        }
    },
    "Indochina": {
        "long": {
            "standard": "Indochina-Zeit"
        }
    },
    "Indonesia_Central": {
        "long": {
            "standard": "Zentralindonesische Zeit"
        }
    },
    "Indonesia_Eastern": {
        "long": {
            "standard": "Ostindonesische Zeit"
        }
    },
    "Indonesia_Western": {
        "long": {
            "standard": "Westindonesische Zeit"
        }
    },
    "Iran": {
        "long": {
            "generic": "Iranische Zeit",
            "standard": "Iranische Normalzeit",
            "daylight": "Iranische Sommerzeit"
        }
    },
    "Irkutsk": {
        "long": {
            "generic": "Irkutsk-Zeit",
            "standard": "Irkutsk-Normalzeit",
            "daylight": "Irkutsk-Sommerzeit"
        }
    },
    "Israel": {
        "long": {
            "generic": "Israelische Zeit",
            "standard": "Israelische Normalzeit",
            "daylight": "Israelische Sommerzeit"
        }
    },
    "Japan": {
        "long": {
            "generic": "Japanische Zeit",
            "standard": "Japanische Normalzeit",
            "daylight": "Japanische Sommerzeit"
        }
    },

```

```
"Kamchatka": {
  "long": {
    "generic": "Kamtschatka-Zeit",
    "standard": "Kamtschatka-Normalzeit",
    "daylight": "Kamtschatka-Sommerzeit"
  }
},
"Kazakhstan_Eastern": {
  "long": {
    "standard": "Ostkasachische Zeit"
  }
},
"Kazakhstan_Western": {
  "long": {
    "standard": "Westkasachische Zeit"
  }
},
"Korea": {
  "long": {
    "generic": "Koreanische Zeit",
    "standard": "Koreanische Normalzeit",
    "daylight": "Koreanische Sommerzeit"
  }
},
"Kosrae": {
  "long": {
    "standard": "Kosrae-Zeit"
  }
},
"Krasnoyarsk": {
  "long": {
    "generic": "Krasnojarsk-Zeit",
    "standard": "Krasnojarsk-Normalzeit",
    "daylight": "Krasnojarsk-Sommerzeit"
  }
},
"Kyrgystan": {
  "long": {
    "standard": "Kirgisistan-Zeit"
  }
},
"Lanka": {
  "long": {
    "standard": "Sri-Lanka-Zeit"
  }
},
"Line_Islands": {
  "long": {
    "standard": "Linieninseln-Zeit"
  }
},
"Lord_Howe": {
  "long": {
    "generic": "Lord-Howe-Zeit",
    "standard": "Lord-Howe-Normalzeit",
    "daylight": "Lord-Howe-Sommerzeit"
  }
}
```

```
    },
    "Macau": {
      "long": {
        "generic": "Macau-Zeit",
        "standard": "Macau-Normalzeit",
        "daylight": "Macau-Sommerzeit"
      }
    },
    "Macquarie": {
      "long": {
        "standard": "Macquarieinsel-Zeit"
      }
    },
    "Magadan": {
      "long": {
        "generic": "Magadan-Zeit",
        "standard": "Magadan-Normalzeit",
        "daylight": "Magadan-Sommerzeit"
      }
    },
    "Malaysia": {
      "long": {
        "standard": "Malaysische Zeit"
      }
    },
    "Maldives": {
      "long": {
        "standard": "Malediven-Zeit"
      }
    },
    "Marquesas": {
      "long": {
        "standard": "Marquesas-Zeit"
      }
    },
    "Marshall_Islands": {
      "long": {
        "standard": "Marshallinseln-Zeit"
      }
    },
    "Mauritius": {
      "long": {
        "generic": "Mauritius-Zeit",
        "standard": "Mauritius-Normalzeit",
        "daylight": "Mauritius-Sommerzeit"
      }
    },
    "Mawson": {
      "long": {
        "standard": "Mawson-Zeit"
      }
    },
    "Mexico_Northwest": {
      "long": {
        "generic": "Mexiko Nordwestliche Zone-Zeit",
        "standard": "Mexiko Nordwestliche Zone-Normalzeit",
        "daylight": "Mexiko Nordwestliche Zone-Sommerzeit"
      }
    }
  },
}
```

```

    }
  },
  "Mexico_Pacific": {
    "long": {
      "generic": "Mexiko Pazifikzone-Zeit",
      "standard": "Mexiko Pazifikzone-Normalzeit",
      "daylight": "Mexiko Pazifikzone-Sommerzeit"
    }
  },
  "Mongolia": {
    "long": {
      "generic": "Ulaanbaatar-Zeit",
      "standard": "Ulaanbaatar-Normalzeit",
      "daylight": "Ulaanbaatar-Sommerzeit"
    }
  },
  "Moscow": {
    "long": {
      "generic": "Moskauer Zeit",
      "standard": "Moskauer Normalzeit",
      "daylight": "Moskauer Sommerzeit"
    }
  },
  "Myanmar": {
    "long": {
      "standard": "Myanmar-Zeit"
    }
  },
  "Nauru": {
    "long": {
      "standard": "Nauru-Zeit"
    }
  },
  "Nepal": {
    "long": {
      "standard": "Nepalesische Zeit"
    }
  },
  "New_Caledonia": {
    "long": {
      "generic": "Neukaledonische Zeit",
      "standard": "Neukaledonische Normalzeit",
      "daylight": "Neukaledonische Sommerzeit"
    }
  },
  "New_Zealand": {
    "long": {
      "generic": "Neuseeland-Zeit",
      "standard": "Neuseeland-Normalzeit",
      "daylight": "Neuseeland-Sommerzeit"
    }
  },
  "Newfoundland": {
    "long": {
      "generic": "Neufundland-Zeit",
      "standard": "Neufundland-Normalzeit",
      "daylight": "Neufundland-Sommerzeit"
    }
  }
}

```

```

    }
  },
  "Niue": {
    "long": {
      "standard": "Niue-Zeit"
    }
  },
  "Norfolk": {
    "long": {
      "standard": "Norfolkinsel-Zeit"
    }
  },
  "Noronha": {
    "long": {
      "generic": "Fernando de Noronha-Zeit",
      "standard": "Fernando de Noronha-Normalzeit",
      "daylight": "Fernando de Noronha-Sommerzeit"
    }
  },
  "North_Mariana": {
    "long": {
      "standard": "Nördliche-Marianen-Zeit"
    }
  },
  "Novosibirsk": {
    "long": {
      "generic": "Nowosibirsk-Zeit",
      "standard": "Nowosibirsk-Normalzeit",
      "daylight": "Nowosibirsk-Sommerzeit"
    }
  },
  "Omsk": {
    "long": {
      "generic": "Omsk-Zeit",
      "standard": "Omsk-Normalzeit",
      "daylight": "Omsk-Sommerzeit"
    }
  },
  "Pakistan": {
    "long": {
      "generic": "Pakistanische Zeit",
      "standard": "Pakistanische Normalzeit",
      "daylight": "Pakistanische Sommerzeit"
    }
  },
  "Palau": {
    "long": {
      "standard": "Palau-Zeit"
    }
  },
  "Papua_New_Guinea": {
    "long": {
      "standard": "Papua-Neuguinea-Zeit"
    }
  },
  "Paraguay": {
    "long": {

```

```

        "generic": "Paraguayische Zeit",
        "standard": "Paraguayische Normalzeit",
        "daylight": "Paraguayische Sommerzeit"
    },
},
"Peru": {
    "long": {
        "generic": "Peruanische Zeit",
        "standard": "Peruanische Normalzeit",
        "daylight": "Peruanische Sommerzeit"
    }
},
"Philippines": {
    "long": {
        "generic": "Philippinische Zeit",
        "standard": "Philippinische Normalzeit",
        "daylight": "Philippinische Sommerzeit"
    }
},
"Phoenix_Islands": {
    "long": {
        "standard": "Phoenixinseln-Zeit"
    }
},
"Pierre_Miquelon": {
    "long": {
        "generic": "Saint-Pierre-und-Miquelon-Zeit",
        "standard": "Saint-Pierre-und-Miquelon-Normalzeit",
        "daylight": "Saint-Pierre-und-Miquelon-Sommerzeit"
    }
},
"Pitcairn": {
    "long": {
        "standard": "Pitcairnsinseln-Zeit"
    }
},
"Ponape": {
    "long": {
        "standard": "Ponape-Zeit"
    }
},
"Pyongyang": {
    "long": {
        "standard": "Pjöngjang-Zeit"
    }
},
"Qyzylorda": {
    "long": {
        "generic": "Quysylorda-Zeit",
        "standard": "Quysylorda-Normalzeit",
        "daylight": "Qysylorda-Sommerzeit"
    }
},
"Reunion": {
    "long": {
        "standard": "Réunion-Zeit"
    }
}

```



```
    },
    "Rothera": {
      "long": {
        "standard": "Rothera-Zeit"
      }
    },
    "Sakhalin": {
      "long": {
        "generic": "Sachalin-Zeit",
        "standard": "Sachalin-Normalzeit",
        "daylight": "Sachalin-Sommerzeit"
      }
    },
    "Samara": {
      "long": {
        "generic": "Samara-Zeit",
        "standard": "Samara-Normalzeit",
        "daylight": "Samara-Sommerzeit"
      }
    },
    "Samoa": {
      "long": {
        "generic": "Samoa-Zeit",
        "standard": "Samoa-Normalzeit",
        "daylight": "Samoa-Sommerzeit"
      }
    },
    "Seychelles": {
      "long": {
        "standard": "Seychellen-Zeit"
      }
    },
    "Singapore": {
      "long": {
        "standard": "Singapur-Zeit"
      }
    },
    "Solomon": {
      "long": {
        "standard": "Salomoninseln-Zeit"
      }
    },
    "South_Georgia": {
      "long": {
        "standard": "Südgeorgische Zeit"
      }
    },
    "Suriname": {
      "long": {
        "standard": "Suriname-Zeit"
      }
    },
    "Syowa": {
      "long": {
        "standard": "Syowa-Zeit"
      }
    },
  },
```

```
"Tahiti": {
  "long": {
    "standard": "Tahiti-Zeit"
  }
},
"Taipei": {
  "long": {
    "generic": "Taipeh-Zeit",
    "standard": "Taipeh-Normalzeit",
    "daylight": "Taipeh-Sommerzeit"
  }
},
"Tajikistan": {
  "long": {
    "standard": "Tadschikistan-Zeit"
  }
},
"Tokelau": {
  "long": {
    "standard": "Tokelau-Zeit"
  }
},
"Tonga": {
  "long": {
    "generic": "Tonganische Zeit",
    "standard": "Tonganische Normalzeit",
    "daylight": "Tonganische Sommerzeit"
  }
},
"Truk": {
  "long": {
    "standard": "Chuuk-Zeit"
  }
},
"Turkmenistan": {
  "long": {
    "generic": "Turkmenistan-Zeit",
    "standard": "Turkmenistan-Normalzeit",
    "daylight": "Turkmenistan-Sommerzeit"
  }
},
"Tuvalu": {
  "long": {
    "standard": "Tuvalu-Zeit"
  }
},
"Uruguay": {
  "long": {
    "generic": "Uruguayanische Zeit",
    "standard": "Uruguayanische Normalzeit",
    "daylight": "Uruguayanische Sommerzeit"
  }
},
"Uzbekistan": {
  "long": {
    "generic": "Usbekistan-Zeit",
    "standard": "Usbekistan-Normalzeit",
```

```
        "daylight": "Usbekistan-Sommerzeit"
      }
    },
    "Vanuatu": {
      "long": {
        "generic": "Vanuatu-Zeit",
        "standard": "Vanuatu-Normalzeit",
        "daylight": "Vanuatu-Sommerzeit"
      }
    },
    "Venezuela": {
      "long": {
        "standard": "Venezuela-Zeit"
      }
    },
    "Vladivostok": {
      "long": {
        "generic": "Wladiwostok-Zeit",
        "standard": "Wladiwostok-Normalzeit",
        "daylight": "Wladiwostok-Sommerzeit"
      }
    },
    "Wolgograd": {
      "long": {
        "generic": "Wolgograd-Zeit",
        "standard": "Wolgograd-Normalzeit",
        "daylight": "Wolgograd-Sommerzeit"
      }
    },
    "Vostok": {
      "long": {
        "standard": "Wostok-Zeit"
      }
    },
    "Wake": {
      "long": {
        "standard": "Wake-Insel-Zeit"
      }
    },
    "Wallis": {
      "long": {
        "standard": "Wallis-und-Futuna-Zeit"
      }
    },
    "Yakutsk": {
      "long": {
        "generic": "Jakutsk-Zeit",
        "standard": "Jakutsk-Normalzeit",
        "daylight": "Jakutsk-Sommerzeit"
      }
    },
    "Yekaterinburg": {
      "long": {
        "generic": "Jekaterinburg-Zeit",
        "standard": "Jekaterinburg-Normalzeit",
        "daylight": "Jekaterinburg-Sommerzeit"
      }
    }
  }
}
```

$$\left\{ \begin{array}{c} \\ \end{array} \right\}$$

TIMEZONENAMES.JSX

```
{
  "main";
  {
    "de";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 12879 $",
          "_cldrVersion";
          "30.0.3";
        }
        "language";
        "de";
      }
      "dates";
      {
        "timeZoneNames";
        {
          "hourFormat";
          "+HH:mm;-HH:mm",
          "gmtFormat";
          "GMT{0}",
          "gmtZeroFormat";
          "GMT",
          "regionFormat";
          "{0} Zeit",
          "regionFormat-type-daylight";
          "{0} Sommerzeit",
          "regionFormat-type-standard";
          "{0} Normalzeit",
          "fallbackFormat";
          "{1} ({0})",
          "zone";
          {
            "America";
            {
              "Adak";
              {
                "exemplarCity";
                "Adak";
              }
              "Anchorage";
              {
```

```
        "exemplarCity";
        "Anchorage";
    }
    "Anguilla";
    {
        "exemplarCity";
        "Anguilla";
    }
    "Antigua";
    {
        "exemplarCity";
        "Antigua";
    }
    "Araguaina";
    {
        "exemplarCity";
        "Araguaina";
    }
    "Argentina";
    {
        "Rio_Gallegos";
        {
            "exemplarCity";
            "Rio Gallegos";
        }
        "San_Juan";
        {
            "exemplarCity";
            "San Juan";
        }
        "Ushuaia";
        {
            "exemplarCity";
            "Ushuaia";
        }
        "La_Rioja";
        {
            "exemplarCity";
            "La Rioja";
        }
        "San_Luis";
        {
            "exemplarCity";
            "San Luis";
        }
        "Salta";
        {
            "exemplarCity";
            "Salta";
        }
        "Tucuman";
        {
            "exemplarCity";
            "Tucuman";
        }
    }
    "Aruba";
```

```
{
    "exemplarCity";
    "Aruba";
}
"Asuncion";
{
    "exemplarCity";
    "Asunción";
}
"Bahia";
{
    "exemplarCity";
    "Bahia";
}
"Bahia_Banderas";
{
    "exemplarCity";
    "Bahia Banderas";
}
"Barbados";
{
    "exemplarCity";
    "Barbados";
}
"Belem";
{
    "exemplarCity";
    "Belem";
}
"Belize";
{
    "exemplarCity";
    "Belize";
}
"Blanc-Sablon";
{
    "exemplarCity";
    "Blanc-Sablon";
}
"Boa_Vista";
{
    "exemplarCity";
    "Boa Vista";
}
"Bogota";
{
    "exemplarCity";
    "Bogotá";
}
"Boise";
{
    "exemplarCity";
    "Boise";
}
"Buenos_Aires";
{
    "exemplarCity";
```

```
        "Buenos Aires";
    }
    "Cambridge_Bay";
    {
        "exemplarCity";
        "Cambridge Bay";
    }
    "Campo_Grande";
    {
        "exemplarCity";
        "Campo Grande";
    }
    "Cancun";
    {
        "exemplarCity";
        "Cancún";
    }
    "Caracas";
    {
        "exemplarCity";
        "Caracas";
    }
    "Catamarca";
    {
        "exemplarCity";
        "Catamarca";
    }
    "Cayenne";
    {
        "exemplarCity";
        "Cayenne";
    }
    "Cayman";
    {
        "exemplarCity";
        "Kaimaninseln";
    }
    "Chicago";
    {
        "exemplarCity";
        "Chicago";
    }
    "Chihuahua";
    {
        "exemplarCity";
        "Chihuahua";
    }
    "Coral_Harbour";
    {
        "exemplarCity";
        "Atikokan";
    }
    "Cordoba";
    {
        "exemplarCity";
        "Córdoba";
    }
}
```

```
"Costa_Rica";
{
  "exemplarCity";
  "Costa Rica";
}
"Creston";
{
  "exemplarCity";
  "Creston";
}
"Cuiaba";
{
  "exemplarCity";
  "Cuiaba";
}
"Curacao";
{
  "exemplarCity";
  "Curaçao";
}
"Danmarkshavn";
{
  "exemplarCity";
  "Danmarkshavn";
}
"Dawson";
{
  "exemplarCity";
  "Dawson";
}
"Dawson_Creek";
{
  "exemplarCity";
  "Dawson Creek";
}
"Denver";
{
  "exemplarCity";
  "Denver";
}
"Detroit";
{
  "exemplarCity";
  "Detroit";
}
"Dominica";
{
  "exemplarCity";
  "Dominica";
}
"Edmonton";
{
  "exemplarCity";
  "Edmonton";
}
"Eirunepe";
{
```



```
        "exemplarCity";
        "Eirunepe";
    }
    "El_Salvador";
    {
        "exemplarCity";
        "El Salvador";
    }
    "Fort_Nelson";
    {
        "exemplarCity";
        "Fort Nelson";
    }
    "Fortaleza";
    {
        "exemplarCity";
        "Fortaleza";
    }
    "Glace_Bay";
    {
        "exemplarCity";
        "Glace Bay";
    }
    "Godthab";
    {
        "exemplarCity";
        "Nuuk";
    }
    "Goose_Bay";
    {
        "exemplarCity";
        "Goose Bay";
    }
    "Grand_Turk";
    {
        "exemplarCity";
        "Grand Turk";
    }
    "Grenada";
    {
        "exemplarCity";
        "Grenada";
    }
    "Guadeloupe";
    {
        "exemplarCity";
        "Guadeloupe";
    }
    "Guatemala";
    {
        "exemplarCity";
        "Guatemala";
    }
    "Guayaquil";
    {
        "exemplarCity";
        "Guayaquil";
    }
```

```
}
"Guyana";
{
  "exemplarCity";
  "Guyana";
}
"Halifax";
{
  "exemplarCity";
  "Halifax";
}
"Havana";
{
  "exemplarCity";
  "Havanna";
}
"Hermosillo";
{
  "exemplarCity";
  "Hermosillo";
}
"Indiana";
{
  "Vincennes";
  {
    "exemplarCity";
    "Vincennes, Indiana";
  }
  "Petersburg";
  {
    "exemplarCity";
    "Petersburg, Indiana";
  }
  "Tell_City";
  {
    "exemplarCity";
    "Tell City, Indiana";
  }
  "Knox";
  {
    "exemplarCity";
    "Knox, Indiana";
  }
  "Winamac";
  {
    "exemplarCity";
    "Winamac, Indiana";
  }
  "Marengo";
  {
    "exemplarCity";
    "Marengo, Indiana";
  }
  "Vevay";
  {
    "exemplarCity";
    "Vevay, Indiana";
  }
}
```

```
    }  
  }  
  "Indianapolis";  
  {  
    "exemplarCity";  
    "Indianapolis";  
  }  
  "Inuvik";  
  {  
    "exemplarCity";  
    "Inuvik";  
  }  
  "Iqaluit";  
  {  
    "exemplarCity";  
    "Iqaluit";  
  }  
  "Jamaica";  
  {  
    "exemplarCity";  
    "Jamaika";  
  }  
  "Jujuy";  
  {  
    "exemplarCity";  
    "Jujuy";  
  }  
  "Juneau";  
  {  
    "exemplarCity";  
    "Juneau";  
  }  
  "Kentucky";  
  {  
    "Monticello";  
    {  
      "exemplarCity";  
      "Monticello, Kentucky";  
    }  
  }  
  "Kralendijk";  
  {  
    "exemplarCity";  
    "Kralendijk";  
  }  
  "La_Paz";  
  {  
    "exemplarCity";  
    "La Paz";  
  }  
  "Lima";  
  {  
    "exemplarCity";  
    "Lima";  
  }  
  "Los_Angeles";  
  {
```

```
        "exemplarCity";
        "Los Angeles";
    }
    "Louisville";
    {
        "exemplarCity";
        "Louisville";
    }
    "Lower_Princes";
    {
        "exemplarCity";
        "Lower Prince's Quarter";
    }
    "Maceio";
    {
        "exemplarCity";
        "Maceio";
    }
    "Managua";
    {
        "exemplarCity";
        "Managua";
    }
    "Manaus";
    {
        "exemplarCity";
        "Manaus";
    }
    "Marigot";
    {
        "exemplarCity";
        "Marigot";
    }
    "Martinique";
    {
        "exemplarCity";
        "Martinique";
    }
    "Matamoros";
    {
        "exemplarCity";
        "Matamoros";
    }
    "Mazatlan";
    {
        "exemplarCity";
        "Mazatlan";
    }
    "Mendoza";
    {
        "exemplarCity";
        "Mendoza";
    }
    "Menominee";
    {
        "exemplarCity";
        "Menominee";
    }
}
```

```
}
"Merida";
{
  "exemplarCity";
  "Merida";
}
"Metlakatla";
{
  "exemplarCity";
  "Metlakatla";
}
"Mexico_City";
{
  "exemplarCity";
  "Mexiko-Stadt";
}
"Miquelon";
{
  "exemplarCity";
  "Miquelon";
}
"Moncton";
{
  "exemplarCity";
  "Moncton";
}
"Monterrey";
{
  "exemplarCity";
  "Monterrey";
}
"Montevideo";
{
  "exemplarCity";
  "Montevideo";
}
"Montserrat";
{
  "exemplarCity";
  "Montserrat";
}
"Nassau";
{
  "exemplarCity";
  "Nassau";
}
"New_York";
{
  "exemplarCity";
  "New York";
}
"Nipigon";
{
  "exemplarCity";
  "Nipigon";
}
"Nome";
```

```
{
  "exemplarCity";
  "Nome";
}
"Noronha";
{
  "exemplarCity";
  "Noronha";
}
"North_Dakota";
{
  "Beulah";
  {
    "exemplarCity";
    "Beulah, North Dakota";
  }
  "New_Salem";
  {
    "exemplarCity";
    "New Salem, North Dakota";
  }
  "Center";
  {
    "exemplarCity";
    "Center, North Dakota";
  }
}
"Ojinaga";
{
  "exemplarCity";
  "Ojinaga";
}
"Panama";
{
  "exemplarCity";
  "Panama";
}
"Pangnirtung";
{
  "exemplarCity";
  "Pangnirtung";
}
"Paramaribo";
{
  "exemplarCity";
  "Paramaribo";
}
"Phoenix";
{
  "exemplarCity";
  "Phoenix";
}
"Port-au-Prince";
{
  "exemplarCity";
  "Port-au-Prince";
}
}
```

```
"Port_of_Spain";
{
  "exemplarCity";
  "Port of Spain";
}
"Porto_Velho";
{
  "exemplarCity";
  "Porto Velho";
}
"Puerto_Rico";
{
  "exemplarCity";
  "Puerto Rico";
}
"Rainy_River";
{
  "exemplarCity";
  "Rainy River";
}
"Rankin_Inlet";
{
  "exemplarCity";
  "Rankin Inlet";
}
"Recife";
{
  "exemplarCity";
  "Recife";
}
"Regina";
{
  "exemplarCity";
  "Regina";
}
"Resolute";
{
  "exemplarCity";
  "Resolute";
}
"Rio_Branco";
{
  "exemplarCity";
  "Rio Branco";
}
"Santa_Isabel";
{
  "exemplarCity";
  "Santa Isabel";
}
"Santarem";
{
  "exemplarCity";
  "Santarem";
}
"Santiago";
{
```

```
        "exemplarCity";
        "Santiago";
    }
    "Santo_Domingo";
    {
        "exemplarCity";
        "Santo Domingo";
    }
    "Sao_Paulo";
    {
        "exemplarCity";
        "São Paulo";
    }
    "Scoresbysund";
    {
        "exemplarCity";
        "Ittoqqortoormiit";
    }
    "Sitka";
    {
        "exemplarCity";
        "Sitka";
    }
    "St_Barthelemy";
    {
        "exemplarCity";
        "Saint-Barthélemy";
    }
    "St_Johns";
    {
        "exemplarCity";
        "St. John's";
    }
    "St_Kitts";
    {
        "exemplarCity";
        "St. Kitts";
    }
    "St_Lucia";
    {
        "exemplarCity";
        "St. Lucia";
    }
    "St_Thomas";
    {
        "exemplarCity";
        "St. Thomas";
    }
    "St_Vincent";
    {
        "exemplarCity";
        "St. Vincent";
    }
    "Swift_Current";
    {
        "exemplarCity";
        "Swift Current";
    }
}
```



```
}
  "Tegucigalpa";
  {
    "exemplarCity";
    "Tegucigalpa";
  }
  "Thule";
  {
    "exemplarCity";
    "Thule";
  }
  "Thunder_Bay";
  {
    "exemplarCity";
    "Thunder Bay";
  }
  "Tijuana";
  {
    "exemplarCity";
    "Tijuana";
  }
  "Toronto";
  {
    "exemplarCity";
    "Toronto";
  }
  "Tortola";
  {
    "exemplarCity";
    "Tortola";
  }
  "Vancouver";
  {
    "exemplarCity";
    "Vancouver";
  }
  "Whitehorse";
  {
    "exemplarCity";
    "Whitehorse";
  }
  "Winnipeg";
  {
    "exemplarCity";
    "Winnipeg";
  }
  "Yakutat";
  {
    "exemplarCity";
    "Yakutat";
  }
  "Yellowknife";
  {
    "exemplarCity";
    "Yellowknife";
  }
}
```

```
"Atlantic";
{
  "Azores";
  {
    "exemplarCity";
    "Azoren";
  }
  "Bermuda";
  {
    "exemplarCity";
    "Bermudas";
  }
  "Canary";
  {
    "exemplarCity";
    "Kanaren";
  }
  "Cape_Verde";
  {
    "exemplarCity";
    "Cabo Verde";
  }
  "Faeroe";
  {
    "exemplarCity";
    "Färöer";
  }
  "Madeira";
  {
    "exemplarCity";
    "Madeira";
  }
  "Reykjavik";
  {
    "exemplarCity";
    "Reykjavík";
  }
  "South_Georgia";
  {
    "exemplarCity";
    "Südgeorgien";
  }
  "St_Helena";
  {
    "exemplarCity";
    "St. Helena";
  }
  "Stanley";
  {
    "exemplarCity";
    "Stanley";
  }
}
"Europe";
{
  "Amsterdam";
  {
```

```
        "exemplarCity";
        "Amsterdam";
    }
    "Andorra";
    {
        "exemplarCity";
        "Andorra";
    }
    "Astrakhan";
    {
        "exemplarCity";
        "Astrachan";
    }
    "Athens";
    {
        "exemplarCity";
        "Athen";
    }
    "Belgrade";
    {
        "exemplarCity";
        "Belgrad";
    }
    "Berlin";
    {
        "exemplarCity";
        "Berlin";
    }
    "Bratislava";
    {
        "exemplarCity";
        "Bratislava";
    }
    "Brussels";
    {
        "exemplarCity";
        "Brüssel";
    }
    "Bucharest";
    {
        "exemplarCity";
        "Bukarest";
    }
    "Budapest";
    {
        "exemplarCity";
        "Budapest";
    }
    "Busingen";
    {
        "exemplarCity";
        "Büsingen";
    }
    "Chisinau";
    {
        "exemplarCity";
        "Kischinau";
    }
```

```
}
"Copenhagen";
{
  "exemplarCity";
  "Kopenhagen";
}
"Dublin";
{
  "long";
  {
    "daylight";
    "Irische Sommerzeit";
  }
  "exemplarCity";
  "Dublin";
}
"Gibraltar";
{
  "exemplarCity";
  "Gibraltar";
}
"Guernsey";
{
  "exemplarCity";
  "Guernsey";
}
"Helsinki";
{
  "exemplarCity";
  "Helsinki";
}
"Isle_of_Man";
{
  "exemplarCity";
  "Isle of Man";
}
"Istanbul";
{
  "exemplarCity";
  "Istanbul";
}
"Jersey";
{
  "exemplarCity";
  "Jersey";
}
"Kaliningrad";
{
  "exemplarCity";
  "Kaliningrad";
}
"Kiev";
{
  "exemplarCity";
  "Kiew";
}
"Kirov";
```

```
{
    "exemplarCity";
    "Kirow";
}
"Lisbon";
{
    "exemplarCity";
    "Lissabon";
}
"Ljubljana";
{
    "exemplarCity";
    "Ljubljana";
}
"London";
{
    "long";
    {
        "daylight";
        "Britische Sommerzeit";
    }
    "exemplarCity";
    "London";
}
"Luxembourg";
{
    "exemplarCity";
    "Luxemburg";
}
"Madrid";
{
    "exemplarCity";
    "Madrid";
}
"Malta";
{
    "exemplarCity";
    "Malta";
}
"Mariehamn";
{
    "exemplarCity";
    "Mariehamn";
}
"Minsk";
{
    "exemplarCity";
    "Minsk";
}
"Monaco";
{
    "exemplarCity";
    "Monaco";
}
"Moscow";
{
    "exemplarCity";
```

```
        "Moskau";
    }
    "Oslo";
    {
        "exemplarCity";
        "Oslo";
    }
    "Paris";
    {
        "exemplarCity";
        "Paris";
    }
    "Podgorica";
    {
        "exemplarCity";
        "Podgorica";
    }
    "Prague";
    {
        "exemplarCity";
        "Prag";
    }
    "Riga";
    {
        "exemplarCity";
        "Riga";
    }
    "Rome";
    {
        "exemplarCity";
        "Rom";
    }
    "Samara";
    {
        "exemplarCity";
        "Samara";
    }
    "San_Marino";
    {
        "exemplarCity";
        "San Marino";
    }
    "Sarajevo";
    {
        "exemplarCity";
        "Sarajevo";
    }
    "Simferopol";
    {
        "exemplarCity";
        "Simferopol";
    }
    "Skopje";
    {
        "exemplarCity";
        "Skopje";
    }
}
```

```
"Sofia";
{
  "exemplarCity";
  "Sofia";
}
"Stockholm";
{
  "exemplarCity";
  "Stockholm";
}
"Tallinn";
{
  "exemplarCity";
  "Tallinn";
}
"Tirane";
{
  "exemplarCity";
  "Tirana";
}
"Ulyanovsk";
{
  "exemplarCity";
  "Uljanowsk";
}
"Uzhgorod";
{
  "exemplarCity";
  "Uschgorod";
}
"Vaduz";
{
  "exemplarCity";
  "Vaduz";
}
"Vatican";
{
  "exemplarCity";
  "Vatikan";
}
"Vienna";
{
  "exemplarCity";
  "Wien";
}
"Vilnius";
{
  "exemplarCity";
  "Vilnius";
}
"Volgograd";
{
  "exemplarCity";
  "Wolgograd";
}
"Warsaw";
{
```

```
        "exemplarCity";
        "Warschau";
    }
    "Zagreb";
    {
        "exemplarCity";
        "Zagreb";
    }
    "Zaporozhye";
    {
        "exemplarCity";
        "Saporischja";
    }
    "Zurich";
    {
        "exemplarCity";
        "Zürich";
    }
}
"Africa";
{
    "Abidjan";
    {
        "exemplarCity";
        "Abidjan";
    }
    "Accra";
    {
        "exemplarCity";
        "Accra";
    }
    "Addis_Ababa";
    {
        "exemplarCity";
        "Addis Abeba";
    }
    "Algiers";
    {
        "exemplarCity";
        "Algier";
    }
    "Asmera";
    {
        "exemplarCity";
        "Asmara";
    }
    "Bamako";
    {
        "exemplarCity";
        "Bamako";
    }
    "Bangui";
    {
        "exemplarCity";
        "Bangui";
    }
    "Banjul";
```



```
{
    "exemplarCity";
    "Banjul";
}
"Bissau";
{
    "exemplarCity";
    "Bissau";
}
"Blantyre";
{
    "exemplarCity";
    "Blantyre";
}
"Brazzaville";
{
    "exemplarCity";
    "Brazzaville";
}
"Bujumbura";
{
    "exemplarCity";
    "Bujumbura";
}
"Cairo";
{
    "exemplarCity";
    "Kairo";
}
"Casablanca";
{
    "exemplarCity";
    "Casablanca";
}
"Ceuta";
{
    "exemplarCity";
    "Ceuta";
}
"Conakry";
{
    "exemplarCity";
    "Conakry";
}
"Dakar";
{
    "exemplarCity";
    "Dakar";
}
"Dar_es_Salaam";
{
    "exemplarCity";
    "Daressalam";
}
"Djibouti";
{
    "exemplarCity";
```

```
        "Dschibuti";
    }
    "Douala";
    {
        "exemplarCity";
        "Douala";
    }
    "El_Aaiun";
    {
        "exemplarCity";
        "El Aaiún";
    }
    "Freetown";
    {
        "exemplarCity";
        "Freetown";
    }
    "Gaborone";
    {
        "exemplarCity";
        "Gaborone";
    }
    "Harare";
    {
        "exemplarCity";
        "Harare";
    }
    "Johannesburg";
    {
        "exemplarCity";
        "Johannesburg";
    }
    "Juba";
    {
        "exemplarCity";
        "Juba";
    }
    "Kampala";
    {
        "exemplarCity";
        "Kampala";
    }
    "Khartoum";
    {
        "exemplarCity";
        "Khartum";
    }
    "Kigali";
    {
        "exemplarCity";
        "Kigali";
    }
    "Kinshasa";
    {
        "exemplarCity";
        "Kinshasa";
    }
}
```

```
"Lagos";
{
  "exemplarCity";
  "Lagos";
}
"Libreville";
{
  "exemplarCity";
  "Libreville";
}
"Lome";
{
  "exemplarCity";
  "Lomé";
}
"Luanda";
{
  "exemplarCity";
  "Luanda";
}
"Lubumbashi";
{
  "exemplarCity";
  "Lubumbashi";
}
"Lusaka";
{
  "exemplarCity";
  "Lusaka";
}
"Malabo";
{
  "exemplarCity";
  "Malabo";
}
"Maputo";
{
  "exemplarCity";
  "Maputo";
}
"Maseru";
{
  "exemplarCity";
  "Maseru";
}
"Mbabane";
{
  "exemplarCity";
  "Mbabane";
}
"Mogadishu";
{
  "exemplarCity";
  "Mogadischu";
}
"Monrovia";
{
```

```
        "exemplarCity";
        "Monrovia";
    }
    "Nairobi";
    {
        "exemplarCity";
        "Nairobi";
    }
    "Ndjamena";
    {
        "exemplarCity";
        "N' Djamena";
    }
    "Niamey";
    {
        "exemplarCity";
        "Niamey";
    }
    "Nouakchott";
    {
        "exemplarCity";
        "Nouakchott";
    }
    "Ouagadougou";
    {
        "exemplarCity";
        "Ouagadougou";
    }
    "Porto-Novo";
    {
        "exemplarCity";
        "Porto Novo";
    }
    "Sao_Tome";
    {
        "exemplarCity";
        "São Tomé";
    }
    "Tripoli";
    {
        "exemplarCity";
        "Tripolis";
    }
    "Tunis";
    {
        "exemplarCity";
        "Tunis";
    }
    "Windhoek";
    {
        "exemplarCity";
        "Windhoek";
    }
}
"Asia";
{
    "Aden";
```

```
{
    "exemplarCity";
    "Aden";
}
"Almaty";
{
    "exemplarCity";
    "Almaty";
}
"Amman";
{
    "exemplarCity";
    "Amman";
}
"Anadyr";
{
    "exemplarCity";
    "Anadyr";
}
"Aqtai";
{
    "exemplarCity";
    "Aqtai";
}
"Aqtobe";
{
    "exemplarCity";
    "Aktobe";
}
"Ashgabat";
{
    "exemplarCity";
    "Aşgabat";
}
"Baghdad";
{
    "exemplarCity";
    "Bagdad";
}
"Bahrain";
{
    "exemplarCity";
    "Bahrain";
}
"Baku";
{
    "exemplarCity";
    "Baku";
}
"Bangkok";
{
    "exemplarCity";
    "Bangkok";
}
"Barnaul";
{
    "exemplarCity";
```

```
        "Barnaul";
    }
    "Beirut";
    {
        "exemplarCity";
        "Beirut";
    }
    "Bishkek";
    {
        "exemplarCity";
        "Bischkek";
    }
    "Brunei";
    {
        "exemplarCity";
        "Brunei";
    }
    "Calcutta";
    {
        "exemplarCity";
        "Kalkutta";
    }
    "Chita";
    {
        "exemplarCity";
        "Tschita";
    }
    "Choibalsan";
    {
        "exemplarCity";
        "Tschoibalsan";
    }
    "Colombo";
    {
        "exemplarCity";
        "Colombo";
    }
    "Damascus";
    {
        "exemplarCity";
        "Damaskus";
    }
    "Dhaka";
    {
        "exemplarCity";
        "Dhaka";
    }
    "Dili";
    {
        "exemplarCity";
        "Dili";
    }
    "Dubai";
    {
        "exemplarCity";
        "Dubai";
    }
}
```

```
"Dushanbe";
{
  "exemplarCity";
  "Duschanbe";
}
"Gaza";
{
  "exemplarCity";
  "Gaza";
}
"Hebron";
{
  "exemplarCity";
  "Hebron";
}
"Hong_Kong";
{
  "exemplarCity";
  "Hongkong";
}
"Hovd";
{
  "exemplarCity";
  "Chowd";
}
"Irkutsk";
{
  "exemplarCity";
  "Irkutsk";
}
"Jakarta";
{
  "exemplarCity";
  "Jakarta";
}
"Jayapura";
{
  "exemplarCity";
  "Jayapura";
}
"Jerusalem";
{
  "exemplarCity";
  "Jerusalem";
}
"Kabul";
{
  "exemplarCity";
  "Kabul";
}
"Kamchatka";
{
  "exemplarCity";
  "Kamtschatka";
}
"Karachi";
{
```

```
        "exemplarCity";
        "Karatschi";
    }
    "Katmandu";
    {
        "exemplarCity";
        "Kathmandu";
    }
    "Khandyga";
    {
        "exemplarCity";
        "Chandyga";
    }
    "Krasnoyarsk";
    {
        "exemplarCity";
        "Krasnojarsk";
    }
    "Kuala_Lumpur";
    {
        "exemplarCity";
        "Kuala Lumpur";
    }
    "Kuching";
    {
        "exemplarCity";
        "Kuching";
    }
    "Kuwait";
    {
        "exemplarCity";
        "Kuwait";
    }
    "Macau";
    {
        "exemplarCity";
        "Macao";
    }
    "Magadan";
    {
        "exemplarCity";
        "Magadan";
    }
    "Makassar";
    {
        "exemplarCity";
        "Makassar";
    }
    "Manila";
    {
        "exemplarCity";
        "Manila";
    }
    "Muscat";
    {
        "exemplarCity";
        "Maskat";
    }
}
```



```
}
"Nicosia";
{
  "exemplarCity";
  "Nikosia";
}
"Novokuznetsk";
{
  "exemplarCity";
  "Nowokuznetsk";
}
"Novosibirsk";
{
  "exemplarCity";
  "Nowosibirsk";
}
"Omsk";
{
  "exemplarCity";
  "Omsk";
}
"Oral";
{
  "exemplarCity";
  "Oral";
}
"Phnom_Penh";
{
  "exemplarCity";
  "Phnom Penh";
}
"Pontianak";
{
  "exemplarCity";
  "Pontianak";
}
"Pyongyang";
{
  "exemplarCity";
  "Pjöngjang";
}
"Qatar";
{
  "exemplarCity";
  "Katar";
}
"Qyzylorda";
{
  "exemplarCity";
  "Qysylorda";
}
"Rangoon";
{
  "exemplarCity";
  "Rangun";
}
"Riyadh";
```

```
{
    "exemplarCity";
    "Riad";
}
"Saigon";
{
    "exemplarCity";
    "Ho-Chi-Minh-Stadt";
}
"Sakhalin";
{
    "exemplarCity";
    "Sachalin";
}
"Samarkand";
{
    "exemplarCity";
    "Samarkand";
}
"Seoul";
{
    "exemplarCity";
    "Seoul";
}
"Shanghai";
{
    "exemplarCity";
    "Shanghai";
}
"Singapore";
{
    "exemplarCity";
    "Singapur";
}
"Srednekolymsk";
{
    "exemplarCity";
    "Srednekolymsk";
}
"Taipei";
{
    "exemplarCity";
    "Taipeh";
}
"Tashkent";
{
    "exemplarCity";
    "Taschkent";
}
"Tbilisi";
{
    "exemplarCity";
    "Tiflis";
}
"Tehran";
{
    "exemplarCity";
```

```
        "Teheran";
    }
    "Thimphu";
    {
        "exemplarCity";
        "Thimphu";
    }
    "Tokyo";
    {
        "exemplarCity";
        "Tokio";
    }
    "Tomsk";
    {
        "exemplarCity";
        "Tomsk";
    }
    "Ulaanbaatar";
    {
        "exemplarCity";
        "Ulaanbaatar";
    }
    "Urumqi";
    {
        "exemplarCity";
        "Ürümqi";
    }
    "Ust-Nera";
    {
        "exemplarCity";
        "Ust-Nera";
    }
    "Vientiane";
    {
        "exemplarCity";
        "Vientiane";
    }
    "Vladivostok";
    {
        "exemplarCity";
        "Wladiwostok";
    }
    "Yakutsk";
    {
        "exemplarCity";
        "Jakutsk";
    }
    "Yekaterinburg";
    {
        "exemplarCity";
        "Jekaterinburg";
    }
    "Yerevan";
    {
        "exemplarCity";
        "Eriwan";
    }
}
```

```
}
"Indian";
{
  "Antananarivo";
  {
    "exemplarCity";
    "Antananarivo";
  }
  "Chagos";
  {
    "exemplarCity";
    "Chagos";
  }
  "Christmas";
  {
    "exemplarCity";
    "Weihnachtsinsel";
  }
  "Cocos";
  {
    "exemplarCity";
    "Cocos";
  }
  "Comoro";
  {
    "exemplarCity";
    "Komoren";
  }
  "Kerguelen";
  {
    "exemplarCity";
    "Kerguelen";
  }
  "Mahe";
  {
    "exemplarCity";
    "Mahe";
  }
  "Maldives";
  {
    "exemplarCity";
    "Malediven";
  }
  "Mauritius";
  {
    "exemplarCity";
    "Mauritius";
  }
  "Mayotte";
  {
    "exemplarCity";
    "Mayotte";
  }
  "Reunion";
  {
    "exemplarCity";
    "Réunion";
  }
}
```

```
    }  
  }  
  "Australia";  
  {  
    "Adelaide";  
    {  
      "exemplarCity";  
      "Adelaide";  
    }  
    "Brisbane";  
    {  
      "exemplarCity";  
      "Brisbane";  
    }  
    "Broken_Hill";  
    {  
      "exemplarCity";  
      "Broken Hill";  
    }  
    "Currie";  
    {  
      "exemplarCity";  
      "Currie";  
    }  
    "Darwin";  
    {  
      "exemplarCity";  
      "Darwin";  
    }  
    "Eucla";  
    {  
      "exemplarCity";  
      "Eucla";  
    }  
    "Hobart";  
    {  
      "exemplarCity";  
      "Hobart";  
    }  
    "Lindeman";  
    {  
      "exemplarCity";  
      "Lindeman";  
    }  
    "Lord_Howe";  
    {  
      "exemplarCity";  
      "Lord Howe";  
    }  
    "Melbourne";  
    {  
      "exemplarCity";  
      "Melbourne";  
    }  
    "Perth";  
    {  
      "exemplarCity";
```

```
        "Perth";
    }
    "Sydney";
    {
        "exemplarCity";
        "Sydney";
    }
}
"Pacific";
{
    "Apia";
    {
        "exemplarCity";
        "Apia";
    }
    "Auckland";
    {
        "exemplarCity";
        "Auckland";
    }
    "Bougainville";
    {
        "exemplarCity";
        "Bougainville";
    }
    "Chatham";
    {
        "exemplarCity";
        "Chatham";
    }
    "Easter";
    {
        "exemplarCity";
        "Osterinsel";
    }
    "Efate";
    {
        "exemplarCity";
        "Efate";
    }
    "Enderbury";
    {
        "exemplarCity";
        "Enderbury";
    }
    "Fakaofo";
    {
        "exemplarCity";
        "Fakaofo";
    }
    "Fiji";
    {
        "exemplarCity";
        "Fidschi";
    }
    "Funafuti";
    {
```

```
        "exemplarCity";
        "Funafuti";
    }
    "Galapagos";
    {
        "exemplarCity";
        "Galapagos";
    }
    "Gambier";
    {
        "exemplarCity";
        "Gambier";
    }
    "Guadalcanal";
    {
        "exemplarCity";
        "Guadalcanal";
    }
    "Guam";
    {
        "exemplarCity";
        "Guam";
    }
    "Honolulu";
    {
        "exemplarCity";
        "Honolulu";
    }
    "Johnston";
    {
        "exemplarCity";
        "Johnston";
    }
    "Kiritimati";
    {
        "exemplarCity";
        "Kiritimati";
    }
    "Kosrae";
    {
        "exemplarCity";
        "Kosrae";
    }
    "Kwajalein";
    {
        "exemplarCity";
        "Kwajalein";
    }
    "Majuro";
    {
        "exemplarCity";
        "Majuro";
    }
    "Marquesas";
    {
        "exemplarCity";
        "Marquesas";
    }
}
```

```
}
"Midway";
{
  "exemplarCity";
  "Midway";
}
"Nauru";
{
  "exemplarCity";
  "Nauru";
}
"Niue";
{
  "exemplarCity";
  "Niue";
}
"Norfolk";
{
  "exemplarCity";
  "Norfolk";
}
"Noumea";
{
  "exemplarCity";
  "Noumea";
}
"Pago_Pago";
{
  "exemplarCity";
  "Pago Pago";
}
"Palau";
{
  "exemplarCity";
  "Palau";
}
"Pitcairn";
{
  "exemplarCity";
  "Pitcairn";
}
"Ponape";
{
  "exemplarCity";
  "Pohnpei";
}
"Port_Moresby";
{
  "exemplarCity";
  "Port Moresby";
}
"Rarotonga";
{
  "exemplarCity";
  "Rarotonga";
}
"Saipan";
```



```
{
    "exemplarCity";
    "Saipan";
}
"Tahiti";
{
    "exemplarCity";
    "Tahiti";
}
"Tarawa";
{
    "exemplarCity";
    "Tarawa";
}
"Tongatapu";
{
    "exemplarCity";
    "Tongatapu";
}
"Truk";
{
    "exemplarCity";
    "Chuuk";
}
"Wake";
{
    "exemplarCity";
    "Wake";
}
"Wallis";
{
    "exemplarCity";
    "Wallis";
}
}
"Arctic";
{
    "Longyearbyen";
    {
        "exemplarCity";
        "Longyearbyen";
    }
}
"Antarctica";
{
    "Casey";
    {
        "exemplarCity";
        "Casey";
    }
    "Davis";
    {
        "exemplarCity";
        "Davis";
    }
    "DumontDUrville";
    {
```

```
        "exemplarCity";
        "Dumont d'Urville";
    }
    "Macquarie";
    {
        "exemplarCity";
        "Macquarie";
    }
    "Mawson";
    {
        "exemplarCity";
        "Mawson";
    }
    "McMurdo";
    {
        "exemplarCity";
        "McMurdo";
    }
    "Palmer";
    {
        "exemplarCity";
        "Palmer";
    }
    "Rothera";
    {
        "exemplarCity";
        "Rothera";
    }
    "Syowa";
    {
        "exemplarCity";
        "Syowa";
    }
    "Troll";
    {
        "exemplarCity";
        "Troll";
    }
    "Vostok";
    {
        "exemplarCity";
        "Wostok";
    }
}
"Etc";
{
    "GMT";
    {
        "exemplarCity";
        "GMT";
    }
    "GMT1";
    {
        "exemplarCity";
        "GMT+1";
    }
    "GMT10";
```

```
{
    "exemplarCity";
    "GMT+10";
}
"GMT11";
{
    "exemplarCity";
    "GMT+11";
}
"GMT12";
{
    "exemplarCity";
    "GMT+12";
}
"GMT2";
{
    "exemplarCity";
    "GMT+2";
}
"GMT3";
{
    "exemplarCity";
    "GMT+3";
}
"GMT4";
{
    "exemplarCity";
    "GMT+4";
}
"GMT5";
{
    "exemplarCity";
    "GMT+5";
}
"GMT6";
{
    "exemplarCity";
    "GMT+6";
}
"GMT7";
{
    "exemplarCity";
    "GMT+7";
}
"GMT8";
{
    "exemplarCity";
    "GMT+8";
}
"GMT9";
{
    "exemplarCity";
    "GMT+9";
}
"GMT-1";
{
    "exemplarCity";
```

```
        "GMT-1";
    }
    "GMT-10";
    {
        "exemplarCity";
        "GMT-10";
    }
    "GMT-11";
    {
        "exemplarCity";
        "GMT-11";
    }
    "GMT-12";
    {
        "exemplarCity";
        "GMT-12";
    }
    "GMT-13";
    {
        "exemplarCity";
        "GMT-13";
    }
    "GMT-14";
    {
        "exemplarCity";
        "GMT-14";
    }
    "GMT-2";
    {
        "exemplarCity";
        "GMT-2";
    }
    "GMT-3";
    {
        "exemplarCity";
        "GMT-3";
    }
    "GMT-4";
    {
        "exemplarCity";
        "GMT-4";
    }
    "GMT-5";
    {
        "exemplarCity";
        "GMT-5";
    }
    "GMT-6";
    {
        "exemplarCity";
        "GMT-6";
    }
    "GMT-7";
    {
        "exemplarCity";
        "GMT-7";
    }
}
```

```

        "GMT-8";
        {
            "exemplarCity";
            "GMT-8";
        }
        "GMT-9";
        {
            "exemplarCity";
            "GMT-9";
        }
        "Unknown";
        {
            "exemplarCity";
            "Unbekannt";
        }
    }
}
"metazone";
{
    "Acre";
    {
        "long";
        {
            "generic";
            "Acre-Zeit",
            "standard";
            "Acre-Normalzeit",
            "daylight";
            "Acre-Sommerzeit";
        }
    }
    "Afghanistan";
    {
        "long";
        {
            "standard";
            "Afghanistan-Zeit";
        }
    }
    "Africa_Central";
    {
        "long";
        {
            "standard";
            "Zentralafrikanische Zeit";
        }
    }
    "Africa_Eastern";
    {
        "long";
        {
            "standard";
            "Ostafrikanische Zeit";
        }
    }
    "Africa_Southern";
    {

```

```

        "long";
        {
            "standard";
            "Südafrikanische Zeit";
        }
    }
    "Africa_Western";
    {
        "long";
        {
            "generic";
            "Westafrikanische Zeit",
            "standard";
            "Westafrikanische Normalzeit",
            "daylight";
            "Westafrikanische Sommerzeit";
        }
    }
    "Alaska";
    {
        "long";
        {
            "generic";
            "Alaska-Zeit",
            "standard";
            "Alaska-Normalzeit",
            "daylight";
            "Alaska-Sommerzeit";
        }
    }
    "Almaty";
    {
        "long";
        {
            "generic";
            "Almaty-Zeit",
            "standard";
            "Almaty-Normalzeit",
            "daylight";
            "Almaty-Sommerzeit";
        }
    }
    "Amazon";
    {
        "long";
        {
            "generic";
            "Amazonas-Zeit",
            "standard";
            "Amazonas-Normalzeit",
            "daylight";
            "Amazonas-Sommerzeit";
        }
    }
    "America_Central";
    {
        "long";

```

```

        {
            "generic";
            "Nordamerikanische Inlandzeit",
            "standard";
            "Nordamerikanische Inland-Normalzeit",
            "daylight";
            "Nordamerikanische Inland-Sommerzeit";
        }
    }
    "America_Eastern";
    {
        "long";
        {
            "generic";
            "Nordamerikanische Ostküstenzeit",
            "standard";
            "Nordamerikanische Ostküsten-Normalzeit",
            "daylight";
            "Nordamerikanische Ostküsten-Sommerzeit";
        }
    }
    "America_Mountain";
    {
        "long";
        {
            "generic";
            "Rocky-Mountain-Zeit",
            "standard";
            "Rocky Mountain-Normalzeit",
            "daylight";
            "Rocky-Mountain-Sommerzeit";
        }
    }
    "America_Pacific";
    {
        "long";
        {
            "generic";
            "Nordamerikanische Westküstenzeit",
            "standard";
            "Nordamerikanische Westküsten-Normalzeit",
            "daylight";
            "Nordamerikanische Westküsten-Sommerzeit";
        }
    }
    "Anadyr";
    {
        "long";
        {
            "generic";
            "Anadyr Zeit",
            "standard";
            "Anadyr Normalzeit",
            "daylight";
            "Anadyr Sommerzeit";
        }
    }
}

```

```
"Apia";
{
  "long";
  {
    "generic";
    "Apia-Zeit",
    "standard";
    "Apia-Normalzeit",
    "daylight";
    "Apia-Sommerzeit";
  }
}
"Aqtau";
{
  "long";
  {
    "generic";
    "Aqtau-Zeit",
    "standard";
    "Aqtau-Normalzeit",
    "daylight";
    "Aqtau-Sommerzeit";
  }
}
"Aqtobe";
{
  "long";
  {
    "generic";
    "Aqtöbe-Zeit",
    "standard";
    "Aqtöbe-Normalzeit",
    "daylight";
    "Aqtöbe-Sommerzeit";
  }
}
"Arabian";
{
  "long";
  {
    "generic";
    "Arabische Zeit",
    "standard";
    "Arabische Normalzeit",
    "daylight";
    "Arabische Sommerzeit";
  }
}
"Argentina";
{
  "long";
  {
    "generic";
    "Argentinische Zeit",
    "standard";
    "Argentinische Normalzeit",
    "daylight";
```



```

        "Argentinische Sommerzeit";
    }
}
"Argentina_Western";
{
    "long";
    {
        "generic";
        "Westargentinische Zeit",
        "standard";
        "Westargentinische Normalzeit",
        "daylight";
        "Westargentinische Sommerzeit";
    }
}
"Armenia";
{
    "long";
    {
        "generic";
        "Armenische Zeit",
        "standard";
        "Armenische Normalzeit",
        "daylight";
        "Armenische Sommerzeit";
    }
}
"Atlantic";
{
    "long";
    {
        "generic";
        "Atlantik-Zeit",
        "standard";
        "Atlantik-Normalzeit",
        "daylight";
        "Atlantik-Sommerzeit";
    }
}
"Australia_Central";
{
    "long";
    {
        "generic";
        "Zentralaustralische Zeit",
        "standard";
        "Zentralaustralische Normalzeit",
        "daylight";
        "Zentralaustralische Sommerzeit";
    }
}
"Australia_CentralWestern";
{
    "long";
    {
        "generic";
        "Zentral-/Westaustralische Zeit",

```

```

        "standard";
        "Zentral-/Westaustralische Normalzeit",
        "daylight";
        "Zentral-/Westaustralische Sommerzeit";
    }
}
"Australia_Eastern";
{
    "long";
    {
        "generic";
        "Ostaustralische Zeit",
        "standard";
        "Ostaustralische Normalzeit",
        "daylight";
        "Ostaustralische Sommerzeit";
    }
}
"Australia_Western";
{
    "long";
    {
        "generic";
        "Westaustralische Zeit",
        "standard";
        "Westaustralische Normalzeit",
        "daylight";
        "Westaustralische Sommerzeit";
    }
}
"Azerbaijan";
{
    "long";
    {
        "generic";
        "Aserbajdschanische Zeit",
        "standard";
        "Aserbajdschanische Normalzeit",
        "daylight";
        "Aserbajdschanische Sommerzeit";
    }
}
"Azores";
{
    "long";
    {
        "generic";
        "Azoren-Zeit",
        "standard";
        "Azoren-Normalzeit",
        "daylight";
        "Azoren-Sommerzeit";
    }
}
"Bangladesh";
{
    "long";

```

```

        {
            "generic";
            "Bangladesch-Zeit",
            "standard";
            "Bangladesch-Normalzeit",
            "daylight";
            "Bangladesch-Sommerzeit";
        }
    }
    "Bhutan";
    {
        "long";
        {
            "standard";
            "Bhutan-Zeit";
        }
    }
    "Bolivia";
    {
        "long";
        {
            "standard";
            "Bolivianische Zeit";
        }
    }
    "Brasilia";
    {
        "long";
        {
            "generic";
            "Brasília-Zeit",
            "standard";
            "Brasília-Normalzeit",
            "daylight";
            "Brasília-Sommerzeit";
        }
    }
    "Brunei";
    {
        "long";
        {
            "standard";
            "Brunei-Zeit";
        }
    }
    "Cape_Verde";
    {
        "long";
        {
            "generic";
            "Cabo-Verde-Zeit",
            "standard";
            "Cabo-Verde-Normalzeit",
            "daylight";
            "Cabo-Verde-Sommerzeit";
        }
    }
}

```

```
"Casey";
{
  "long";
  {
    "standard";
    "Casey-Zeit";
  }
}
"Chamorro";
{
  "long";
  {
    "standard";
    "Chamorro-Zeit";
  }
}
"Chatham";
{
  "long";
  {
    "generic";
    "Chatham-Zeit",
    "standard";
    "Chatham-Normalzeit",
    "daylight";
    "Chatham-Sommerzeit";
  }
}
"Chile";
{
  "long";
  {
    "generic";
    "Chilenische Zeit",
    "standard";
    "Chilenische Normalzeit",
    "daylight";
    "Chilenische Sommerzeit";
  }
}
"China";
{
  "long";
  {
    "generic";
    "Chinesische Zeit",
    "standard";
    "Chinesische Normalzeit",
    "daylight";
    "Chinesische Sommerzeit";
  }
}
"Choibalsan";
{
  "long";
  {
    "generic";
```

```

        "Tschoibalsan-Zeit",
        "standard";
        "Tschoibalsan-Normalzeit",
        "daylight";
        "Tschoibalsan-Sommerzeit";
    }
}
"Christmas";
{
    "long";
    {
        "standard";
        "Weihnachtsinsel-Zeit";
    }
}
"Cocos";
{
    "long";
    {
        "standard";
        "Kokosinseln-Zeit";
    }
}
"Colombia";
{
    "long";
    {
        "generic";
        "Kolumbianische Zeit",
        "standard";
        "Kolumbianische Normalzeit",
        "daylight";
        "Kolumbianische Sommerzeit";
    }
}
"Cook";
{
    "long";
    {
        "generic";
        "Cookinseln-Zeit",
        "standard";
        "Cookinseln-Normalzeit",
        "daylight";
        "Cookinseln-Sommerzeit";
    }
}
"Cuba";
{
    "long";
    {
        "generic";
        "Kubanische Zeit",
        "standard";
        "Kubanische Normalzeit",
        "daylight";
        "Kubanische Sommerzeit";
    }
}

```

```

    }
  }
  "Davis";
  {
    "long";
    {
      "standard";
      "Davis-Zeit";
    }
  }
  "DumontDUrville";
  {
    "long";
    {
      "standard";
      "Dumont-d'Urville-Zeit";
    }
  }
  "East_Timor";
  {
    "long";
    {
      "standard";
      "Osttimor-Zeit";
    }
  }
  "Easter";
  {
    "long";
    {
      "generic";
      "Osterinsel-Zeit",
      "standard";
      "Osterinsel-Normalzeit",
      "daylight";
      "Osterinsel-Sommerzeit";
    }
  }
  "Ecuador";
  {
    "long";
    {
      "standard";
      "Ecuadorianische Zeit";
    }
  }
  "Europe_Central";
  {
    "long";
    {
      "generic";
      "Mittleuropäische Zeit",
      "standard";
      "Mittleuropäische Normalzeit",
      "daylight";
      "Mittleuropäische Sommerzeit";
    }
  }

```

```
        "short";
        {
            "generic";
            "MEZ",
            "standard";
            "MEZ",
            "daylight";
            "MESZ";
        }
    }
    "Europe_Eastern";
    {
        "long";
        {
            "generic";
            "Osteuropäische Zeit",
            "standard";
            "Osteuropäische Normalzeit",
            "daylight";
            "Osteuropäische Sommerzeit";
        }
        "short";
        {
            "generic";
            "OEZ",
            "standard";
            "OEZ",
            "daylight";
            "OESZ";
        }
    }
    "Europe_Further_Eastern";
    {
        "long";
        {
            "standard";
            "Kaliningrader Zeit";
        }
    }
    "Europe_Western";
    {
        "long";
        {
            "generic";
            "Westeuropäische Zeit",
            "standard";
            "Westeuropäische Normalzeit",
            "daylight";
            "Westeuropäische Sommerzeit";
        }
        "short";
        {
            "generic";
            "WEZ",
            "standard";
            "WEZ",
            "daylight";
        }
    }
}
```

```

        "WESZ";
    }
}
"Falkland";
{
    "long";
    {
        "generic";
        "Falklandinseln-Zeit",
        "standard";
        "Falklandinseln-Normalzeit",
        "daylight";
        "Falklandinseln-Sommerzeit";
    }
}
"Fiji";
{
    "long";
    {
        "generic";
        "Fidschi-Zeit",
        "standard";
        "Fidschi-Normalzeit",
        "daylight";
        "Fidschi-Sommerzeit";
    }
}
"French_Guiana";
{
    "long";
    {
        "standard";
        "Französisch-Guayana-Zeit";
    }
}
"French_Southern";
{
    "long";
    {
        "standard";
        "Französische Süd- und Antarktisgebiete-
Zeit";
    }
}
"Galapagos";
{
    "long";
    {
        "standard";
        "Galapagos-Zeit";
    }
}
"Gambier";
{
    "long";
    {
        "standard";

```



```

        "Gambier-Zeit";
    }
}
"Georgia";
{
    "long";
    {
        "generic";
        "Georgische Zeit",
        "standard";
        "Georgische Normalzeit",
        "daylight";
        "Georgische Sommerzeit";
    }
}
"Gilbert_Islands";
{
    "long";
    {
        "standard";
        "Gilbert-Inseln-Zeit";
    }
}
"GMT";
{
    "long";
    {
        "standard";
        "Mittlere Greenwich-Zeit";
    }
}
"Greenland_Eastern";
{
    "long";
    {
        "generic";
        "Ostgrönland-Zeit",
        "standard";
        "Ostgrönland-Normalzeit",
        "daylight";
        "Ostgrönland-Sommerzeit";
    }
}
"Greenland_Western";
{
    "long";
    {
        "generic";
        "Westgrönland-Zeit",
        "standard";
        "Westgrönland-Normalzeit",
        "daylight";
        "Westgrönland-Sommerzeit";
    }
}
"Guam";
{

```

```
        "long";
        {
            "standard";
            "Guam-Zeit";
        }
    }
    "Gulf";
    {
        "long";
        {
            "standard";
            "Golf-Zeit";
        }
    }
    "Guyana";
    {
        "long";
        {
            "standard";
            "Guyana-Zeit";
        }
    }
    "Hawaii_Aleutian";
    {
        "long";
        {
            "generic";
            "Hawaii-Aleuten-Zeit",
            "standard";
            "Hawaii-Aleuten-Normalzeit",
            "daylight";
            "Hawaii-Aleuten-Sommerzeit";
        }
    }
    "Hong_Kong";
    {
        "long";
        {
            "generic";
            "Hongkong-Zeit",
            "standard";
            "Hongkong-Normalzeit",
            "daylight";
            "Hongkong-Sommerzeit";
        }
    }
    "Hovd";
    {
        "long";
        {
            "generic";
            "Chowd-Zeit",
            "standard";
            "Chowd-Normalzeit",
            "daylight";
            "Chowd-Sommerzeit";
        }
    }
```

```
}
"India";
{
  "long";
  {
    "standard";
    "Indische Zeit";
  }
}
"Indian_Ocean";
{
  "long";
  {
    "standard";
    "Indischer Ozean-Zeit";
  }
}
"Indochina";
{
  "long";
  {
    "standard";
    "Indochina-Zeit";
  }
}
"Indonesia_Central";
{
  "long";
  {
    "standard";
    "Zentralindonesische Zeit";
  }
}
"Indonesia_Eastern";
{
  "long";
  {
    "standard";
    "Ostindonesische Zeit";
  }
}
"Indonesia_Western";
{
  "long";
  {
    "standard";
    "Westindonesische Zeit";
  }
}
"Iran";
{
  "long";
  {
    "generic";
    "Iranische Zeit",
    "standard";
    "Iranische Normalzeit",
```

```

        "daylight";
        "Iranische Sommerzeit";
    }
}
"Irkutsk";
{
    "long";
    {
        "generic";
        "Irkutsk-Zeit",
        "standard";
        "Irkutsk-Normalzeit",
        "daylight";
        "Irkutsk-Sommerzeit";
    }
}
"Israel";
{
    "long";
    {
        "generic";
        "Israelische Zeit",
        "standard";
        "Israelische Normalzeit",
        "daylight";
        "Israelische Sommerzeit";
    }
}
"Japan";
{
    "long";
    {
        "generic";
        "Japanische Zeit",
        "standard";
        "Japanische Normalzeit",
        "daylight";
        "Japanische Sommerzeit";
    }
}
"Kamchatka";
{
    "long";
    {
        "generic";
        "Kamtschatka-Zeit",
        "standard";
        "Kamtschatka-Normalzeit",
        "daylight";
        "Kamtschatka-Sommerzeit";
    }
}
"Kazakhstan_Eastern";
{
    "long";
    {
        "standard";
    }
}

```

```
        "Ostkasachische Zeit";
    }
}
"Kazakhstan_Western";
{
    "long";
    {
        "standard";
        "Westkasachische Zeit";
    }
}
"Korea";
{
    "long";
    {
        "generic";
        "Koreanische Zeit",
        "standard";
        "Koreanische Normalzeit",
        "daylight";
        "Koreanische Sommerzeit";
    }
}
"Kosrae";
{
    "long";
    {
        "standard";
        "Kosrae-Zeit";
    }
}
"Krasnoyarsk";
{
    "long";
    {
        "generic";
        "Krasnojarsk-Zeit",
        "standard";
        "Krasnojarsk-Normalzeit",
        "daylight";
        "Krasnojarsk-Sommerzeit";
    }
}
"Kyrgystan";
{
    "long";
    {
        "standard";
        "Kirgisistan-Zeit";
    }
}
"Lanka";
{
    "long";
    {
        "standard";
        "Sri-Lanka-Zeit";
    }
}
```

```

    }
  }
  "Line_Islands";
  {
    "long";
    {
      "standard";
      "Linieninseln-Zeit";
    }
  }
  "Lord_Howe";
  {
    "long";
    {
      "generic";
      "Lord-Howe-Zeit",
      "standard";
      "Lord-Howe-Normalzeit",
      "daylight";
      "Lord-Howe-Sommerzeit";
    }
  }
  "Macau";
  {
    "long";
    {
      "generic";
      "Macau-Zeit",
      "standard";
      "Macau-Normalzeit",
      "daylight";
      "Macau-Sommerzeit";
    }
  }
  "Macquarie";
  {
    "long";
    {
      "standard";
      "Macquarieinsel-Zeit";
    }
  }
  "Magadan";
  {
    "long";
    {
      "generic";
      "Magadan-Zeit",
      "standard";
      "Magadan-Normalzeit",
      "daylight";
      "Magadan-Sommerzeit";
    }
  }
  "Malaysia";
  {
    "long";

```

```
        {
            "standard";
            "Malaysische Zeit";
        }
    }
    "Maldives";
    {
        "long";
        {
            "standard";
            "Malediven-Zeit";
        }
    }
    "Marquesas";
    {
        "long";
        {
            "standard";
            "Marquesas-Zeit";
        }
    }
    "Marshall_Islands";
    {
        "long";
        {
            "standard";
            "Marshallinseln-Zeit";
        }
    }
    "Mauritius";
    {
        "long";
        {
            "generic";
            "Mauritius-Zeit",
            "standard";
            "Mauritius-Normalzeit",
            "daylight";
            "Mauritius-Sommerzeit";
        }
    }
    "Mawson";
    {
        "long";
        {
            "standard";
            "Mawson-Zeit";
        }
    }
    "Mexico_Northwest";
    {
        "long";
        {
            "generic";
            "Mexiko Nordwestliche Zone-Zeit",
            "standard";
            "Mexiko Nordwestliche Zone-Normalzeit",
```

```
        "daylight";
        "Mexiko Nordwestliche Zone-Sommerzeit";
    }
}
"Mexico_Pacific";
{
    "long";
    {
        "generic";
        "Mexiko Pazifikzone-Zeit",
        "standard";
        "Mexiko Pazifikzone-Normalzeit",
        "daylight";
        "Mexiko Pazifikzone-Sommerzeit";
    }
}
"Mongolia";
{
    "long";
    {
        "generic";
        "Ulaanbaatar-Zeit",
        "standard";
        "Ulaanbaatar-Normalzeit",
        "daylight";
        "Ulaanbaatar-Sommerzeit";
    }
}
"Moscow";
{
    "long";
    {
        "generic";
        "Moskauer Zeit",
        "standard";
        "Moskauer Normalzeit",
        "daylight";
        "Moskauer Sommerzeit";
    }
}
"Myanmar";
{
    "long";
    {
        "standard";
        "Myanmar-Zeit";
    }
}
"Nauru";
{
    "long";
    {
        "standard";
        "Nauru-Zeit";
    }
}
"Nepal";
```



```
{
  "long";
  {
    "standard";
    "Nepalesische Zeit";
  }
}
"New_Caledonia";
{
  "long";
  {
    "generic";
    "Neukaledonische Zeit",
    "standard";
    "Neukaledonische Normalzeit",
    "daylight";
    "Neukaledonische Sommerzeit";
  }
}
"New_Zealand";
{
  "long";
  {
    "generic";
    "Neuseeland-Zeit",
    "standard";
    "Neuseeland-Normalzeit",
    "daylight";
    "Neuseeland-Sommerzeit";
  }
}
"Newfoundland";
{
  "long";
  {
    "generic";
    "Neufundland-Zeit",
    "standard";
    "Neufundland-Normalzeit",
    "daylight";
    "Neufundland-Sommerzeit";
  }
}
"Niue";
{
  "long";
  {
    "standard";
    "Niue-Zeit";
  }
}
"Norfolk";
{
  "long";
  {
    "standard";
    "Norfolkinsel-Zeit";
  }
}
```

```
    }  
  }  
  "Noronha";  
  {  
    "long";  
    {  
      "generic";  
      "Fernando de Noronha-Zeit",  
      "standard";  
      "Fernando de Noronha-Normalzeit",  
      "daylight";  
      "Fernando de Noronha-Sommerzeit";  
    }  
  }  
  "North_Mariana";  
  {  
    "long";  
    {  
      "standard";  
      "Nördliche-Marianen-Zeit";  
    }  
  }  
  "Novosibirsk";  
  {  
    "long";  
    {  
      "generic";  
      "Nowosibirsk-Zeit",  
      "standard";  
      "Nowosibirsk-Normalzeit",  
      "daylight";  
      "Nowosibirsk-Sommerzeit";  
    }  
  }  
  "Omsk";  
  {  
    "long";  
    {  
      "generic";  
      "Omsk-Zeit",  
      "standard";  
      "Omsk-Normalzeit",  
      "daylight";  
      "Omsk-Sommerzeit";  
    }  
  }  
  "Pakistan";  
  {  
    "long";  
    {  
      "generic";  
      "Pakistanische Zeit",  
      "standard";  
      "Pakistanische Normalzeit",  
      "daylight";  
      "Pakistanische Sommerzeit";  
    }  
  }
```

```
}
"Palau";
{
  "long";
  {
    "standard";
    "Palau-Zeit";
  }
}
"Papua_New_Guinea";
{
  "long";
  {
    "standard";
    "Papua-Neuguinea-Zeit";
  }
}
"Paraguay";
{
  "long";
  {
    "generic";
    "Paraguayansische Zeit",
    "standard";
    "Paraguayansische Normalzeit",
    "daylight";
    "Paraguayansische Sommerzeit";
  }
}
"Peru";
{
  "long";
  {
    "generic";
    "Peruanische Zeit",
    "standard";
    "Peruanische Normalzeit",
    "daylight";
    "Peruanische Sommerzeit";
  }
}
"Philippines";
{
  "long";
  {
    "generic";
    "Philippinische Zeit",
    "standard";
    "Philippinische Normalzeit",
    "daylight";
    "Philippinische Sommerzeit";
  }
}
"Phoenix_Islands";
{
  "long";
  {
```

```

        "standard";
        "Phoenixinseln-Zeit";
    }
}
"Pierre_Miquelon";
{
    "long";
    {
        "generic";
        "Saint-Pierre-und-Miquelon-Zeit",
        "standard";
        "Saint-Pierre-und-Miquelon-Normalzeit",
        "daylight";
        "Saint-Pierre-und-Miquelon-Sommerzeit";
    }
}
"Pitcairn";
{
    "long";
    {
        "standard";
        "Pitcairninseln-Zeit";
    }
}
"Ponape";
{
    "long";
    {
        "standard";
        "Ponape-Zeit";
    }
}
"Pyongyang";
{
    "long";
    {
        "standard";
        "Pjöngjang-Zeit";
    }
}
"Qyzylorda";
{
    "long";
    {
        "generic";
        "Quysylorda-Zeit",
        "standard";
        "Quysylorda-Normalzeit",
        "daylight";
        "Qysylorda-Sommerzeit";
    }
}
"Reunion";
{
    "long";
    {
        "standard";

```

```
        "Réunion-Zeit";
    }
}
"Rothera";
{
    "long";
    {
        "standard";
        "Rothera-Zeit";
    }
}
"Sakhalin";
{
    "long";
    {
        "generic";
        "Sachalin-Zeit",
        "standard";
        "Sachalin-Normalzeit",
        "daylight";
        "Sachalin-Sommerzeit";
    }
}
"Samara";
{
    "long";
    {
        "generic";
        "Samara-Zeit",
        "standard";
        "Samara-Normalzeit",
        "daylight";
        "Samara-Sommerzeit";
    }
}
"Samoa";
{
    "long";
    {
        "generic";
        "Samoa-Zeit",
        "standard";
        "Samoa-Normalzeit",
        "daylight";
        "Samoa-Sommerzeit";
    }
}
"Seychelles";
{
    "long";
    {
        "standard";
        "Seychellen-Zeit";
    }
}
"Singapore";
{
```

```
        "long";
        {
            "standard";
            "Singapur-Zeit";
        }
    }
    "Solomon";
    {
        "long";
        {
            "standard";
            "Salomoninseln-Zeit";
        }
    }
    "South_Georgia";
    {
        "long";
        {
            "standard";
            "Südgeorgische Zeit";
        }
    }
    "Suriname";
    {
        "long";
        {
            "standard";
            "Suriname-Zeit";
        }
    }
    "Syowa";
    {
        "long";
        {
            "standard";
            "Syowa-Zeit";
        }
    }
    "Tahiti";
    {
        "long";
        {
            "standard";
            "Tahiti-Zeit";
        }
    }
    "Taipei";
    {
        "long";
        {
            "generic";
            "Taipeh-Zeit",
            "standard";
            "Taipeh-Normalzeit",
            "daylight";
            "Taipeh-Sommerzeit";
        }
    }
}
```

```
}
"Tajikistan";
{
  "long";
  {
    "standard";
    "Tadschikistan-Zeit";
  }
}
"Tokelau";
{
  "long";
  {
    "standard";
    "Tokelau-Zeit";
  }
}
"Tonga";
{
  "long";
  {
    "generic";
    "Tonganische Zeit",
    "standard";
    "Tonganische Normalzeit",
    "daylight";
    "Tonganische Sommerzeit";
  }
}
"Truk";
{
  "long";
  {
    "standard";
    "Chuuk-Zeit";
  }
}
"Turkmenistan";
{
  "long";
  {
    "generic";
    "Turkmenistan-Zeit",
    "standard";
    "Turkmenistan-Normalzeit",
    "daylight";
    "Turkmenistan-Sommerzeit";
  }
}
"Tuvalu";
{
  "long";
  {
    "standard";
    "Tuvalu-Zeit";
  }
}
```

```
"Uruguay";
{
  "long";
  {
    "generic";
    "Uruguayanische Zeit",
    "standard";
    "Uruguayanische Normalzeit",
    "daylight";
    "Uruguayanische Sommerzeit";
  }
}
"Uzbekistan";
{
  "long";
  {
    "generic";
    "Usbekistan-Zeit",
    "standard";
    "Usbekistan-Normalzeit",
    "daylight";
    "Usbekistan-Sommerzeit";
  }
}
"Vanuatu";
{
  "long";
  {
    "generic";
    "Vanuatu-Zeit",
    "standard";
    "Vanuatu-Normalzeit",
    "daylight";
    "Vanuatu-Sommerzeit";
  }
}
"Venezuela";
{
  "long";
  {
    "standard";
    "Venezuela-Zeit";
  }
}
"Vladivostok";
{
  "long";
  {
    "generic";
    "Wladiwostok-Zeit",
    "standard";
    "Wladiwostok-Normalzeit",
    "daylight";
    "Wladiwostok-Sommerzeit";
  }
}
"Volgograd";
```



```
{
  "long";
  {
    "generic";
    "Wolgograd-Zeit",
    "standard";
    "Wolgograd-Normalzeit",
    "daylight";
    "Wolgograd-Sommerzeit";
  }
}
"Vostok";
{
  "long";
  {
    "standard";
    "Wostok-Zeit";
  }
}
"Wake";
{
  "long";
  {
    "standard";
    "Wake-Insel-Zeit";
  }
}
"Wallis";
{
  "long";
  {
    "standard";
    "Wallis-und-Futuna-Zeit";
  }
}
"Yakutsk";
{
  "long";
  {
    "generic";
    "Jakutsk-Zeit",
    "standard";
    "Jakutsk-Normalzeit",
    "daylight";
    "Jakutsk-Sommerzeit";
  }
}
"Yekaterinburg";
{
  "long";
  {
    "generic";
    "Jekaterinburg-Zeit",
    "standard";
    "Jekaterinburg-Normalzeit",
    "daylight";
    "Jekaterinburg-Sommerzeit";
  }
}
```

```
}  
}  
}  
}  
}  
}  
}
```

Right-To-Left

The DatePicker supports right-to-left functionality for languages like Arabic, Hebrew to displays the text in the right-to-left direction. Use [enableRtl](#) property to set the RTL direction.

The below code example demonstrates the DatePicker component in Hebrew culture and also explains how to set the localized text to the placeholder using [load](#) method of

L10n class.

[Class-component]

CA-GREGORIAN.JSON

```
{
  "main": {
    "he": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31.0.1"
        },
        "language": "he"
      },
      "dates": {
        "calendars": {
          "gregorian": {
            "months": {
              "format": {
                "abbreviated": {
                  "1": "ינו'",
                  "2": "פבר'",
                  "3": "מרץ",
                  "4": "אפר'",
                  "5": "מאי",
                  "6": "יוני",
                  "7": "יולי",
                  "8": "אוג'",
                  "9": "ספט'",
                  "10": "אוק'",
                  "11": "נוב'",
                  "12": "דצמ'"
                },
                "narrow": {
                  "1": "1",
                  "2": "2",
                  "3": "3",
                  "4": "4",

```

```

        "5": "5",
        "6": "6",
        "7": "7",
        "8": "8",
        "9": "9",
        "10": "10",
        "11": "11",
        "12": "12"
      },
      "wide": {
        "1": "ינואר",
        "2": "פברואר",
        "3": "מרץ",
        "4": "אפריל",
        "5": "מאי",
        "6": "יוני",
        "7": "יולי",
        "8": "אוגוסט",
        "9": "ספטמבר",
        "10": "אוקטובר",
        "11": "נובמבר",
        "12": "דצמבר"
      }
    },
    "stand-alone": {
      "abbreviated": {
        "1": "ינו'",
        "2": "פבר'",
        "3": "מרץ",
        "4": "אפר'",
        "5": "מאי",
        "6": "יוני",
        "7": "יולי",
        "8": "אוג'",
        "9": "ספט'",
        "10": "אוק'",
        "11": "נוב'",
        "12": "דצמ'"
      },
      "narrow": {
        "1": "1",
        "2": "2",
        "3": "3",
        "4": "4",
        "5": "5",
        "6": "6",
        "7": "7",
        "8": "8",
        "9": "9",
        "10": "10",
        "11": "11",
        "12": "12"
      },
      "wide": {
        "1": "ינואר",
        "2": "פברואר",
        "3": "מרץ",

```

```

        "4": "אפריל",
        "5": "מאי",
        "6": "יוני",
        "7": "יולי",
        "8": "אוגוסט",
        "9": "ספטמבר",
        "10": "אוקטובר",
        "11": "נובמבר",
        "12": "דצמבר"
    }
}
},
"days": {
    "format": {
        "abbreviated": {
            "sun": "א' יום",
            "mon": "ב' יום",
            "tue": "ג' יום",
            "wed": "ד' יום",
            "thu": "ה' יום",
            "fri": "ו' יום",
            "sat": "שבת"
        },
        "narrow": {
            "sun": "א'",
            "mon": "ב'",
            "tue": "ג'",
            "wed": "ד'",
            "thu": "ה'",
            "fri": "ו'",
            "sat": "ש'"
        },
        "short": {
            "sun": "א",
            "mon": "ב",
            "tue": "ג",
            "wed": "ד",
            "thu": "ה",
            "fri": "ו",
            "sat": "ש"
        },
        "wide": {
            "sun": "יום ראשון",
            "mon": "יום שני",
            "tue": "יום שלישי",
            "wed": "יום רביעי",
            "thu": "יום חמישי",
            "fri": "יום שישי",
            "sat": "יום שבת"
        }
    },
    "stand-alone": {
        "abbreviated": {
            "sun": "א' יום",
            "mon": "ב' יום",
            "tue": "ג' יום",
            "wed": "ד' יום",

```

```

        "thu": "יום ה'",
        "fri": "יום ו'",
        "sat": "שבת"
    },
    "narrow": {
        "sun": "א'",
        "mon": "ב'",
        "tue": "ג'",
        "wed": "ד'",
        "thu": "ה'",
        "fri": "ו'",
        "sat": "ש'"
    },
    "short": {
        "sun": "א",
        "mon": "ב",
        "tue": "ג",
        "wed": "ד",
        "thu": "ה",
        "fri": "ו",
        "sat": "ש"
    },
    "wide": {
        "sun": "יום ראשון",
        "mon": "יום שני",
        "tue": "יום שלישי",
        "wed": "יום רביעי",
        "thu": "יום חמישי",
        "fri": "יום שישי",
        "sat": "יום שבת"
    }
  },
  "quarters": {
    "format": {
      "abbreviated": {
        "1": "Q1",
        "2": "Q2",
        "3": "Q3",
        "4": "Q4"
      },
      "narrow": {
        "1": "1",
        "2": "2",
        "3": "3",
        "4": "4"
      },
      "wide": {
        "1": "1 רבעון",
        "2": "2 רבעון",
        "3": "3 רבעון",
        "4": "4 רבעון"
      }
    },
    "stand-alone": {
      "abbreviated": {
        "1": "Q1",

```

```

        "2": "Q2",
        "3": "Q3",
        "4": "Q4"
    },
    "narrow": {
        "1": "1",
        "2": "2",
        "3": "3",
        "4": "4"
    },
    "wide": {
        "1": "1 רבעון",
        "2": "2 רבעון",
        "3": "3 רבעון",
        "4": "4 רבעון"
    }
    },
    "dayPeriods": {
        "format": {
            "abbreviated": {
                "midnight": "חצות",
                "am": "לפנה״צ",
                "pm": "אחה״צ",
                "morning1": "בוקר",
                "afternoon1": "צהריים",
                "afternoon2": "אחר הצהריים",
                "evening1": "ערב",
                "night1": "לילה",
                "night2": "לפנות בוקר"
            },
            "narrow": {
                "midnight": "חצות",
                "am": "לפנה״צ",
                "pm": "אחה״צ",
                "morning1": "בוקר",
                "afternoon1": "צהריים",
                "afternoon2": "אחר הצהריים",
                "evening1": "ערב",
                "night1": "לילה",
                "night2": "לפנות בוקר"
            },
            "wide": {
                "midnight": "חצות",
                "am": "לפנה״צ",
                "pm": "אחה״צ",
                "morning1": "בוקר",
                "afternoon1": "צהריים",
                "afternoon2": "אחר הצהריים",
                "evening1": "ערב",
                "night1": "לילה",
                "night2": "לפנות בוקר"
            }
        },
        "stand-alone": {
            "abbreviated": {
                "midnight": "חצות",

```

```

        "am": "לפנה"צ",
        "pm": "אחה"צ",
        "morning1": "בוקר",
        "afternoon1": "צהריים",
        "afternoon2": "אחר הצהריים",
        "evening1": "ערב",
        "night1": "לילה",
        "night2": "לפנות בוקר"
    },
    "narrow": {
        "midnight": "חצות",
        "am": "לפנה"צ",
        "pm": "אחה"צ",
        "morning1": "בוקר",
        "afternoon1": "צהריים",
        "afternoon2": "אחר הצהריים",
        "evening1": "ערב",
        "night1": "לילה",
        "night2": "לפנות בוקר"
    },
    "wide": {
        "midnight": "חצות",
        "am": "לפנה"צ",
        "pm": "אחה"צ",
        "morning1": "בוקר",
        "afternoon1": "צהריים",
        "afternoon2": "אחר הצהריים",
        "evening1": "ערב",
        "night1": "לילה",
        "night2": "לפנות בוקר"
    }
  },
  "eras": {
    "eraNames": {
      "0": "לפני הספירה",
      "0-alt-variant": "לפנה"ס",
      "1": "לספירה",
      "1-alt-variant": "CE"
    },
    "eraAbbr": {
      "0": "לפנה"ס",
      "0-alt-variant": "BCE",
      "1": "לספירה",
      "1-alt-variant": "CE"
    },
    "eraNarrow": {
      "0": "לפנה"ס",
      "0-alt-variant": "BCE",
      "1": "לספירה",
      "1-alt-variant": "CE"
    }
  },
  "dateFormats": {
    "full": "EEEE, d MMMM y",
    "long": "d MMMM y",
    "medium": "d MMM y",

```

```

    "short": "d.M.y"
  },
  "timeFormats": {
    "full": "H:mm:ss zzzz",
    "long": "H:mm:ss z",
    "medium": "H:mm:ss",
    "short": "H:mm"
  },
  "dateTimeFormats": {
    "full": "{1} בשעה {0}",
    "long": "{1} בשעה {0}",
    "medium": "{1}, {0}",
    "short": "{1}, {0}",
    "availableFormats": {
      "d": "d",
      "E": "ccc",
      "Ed": "E ה-d",
      "Ehm": "E h:mm a",
      "EHm": "E H:mm",
      "Ehms": "E h:mm:ss a",
      "EHms": "E H:mm:ss",
      "Gy": "y G",
      "GyMMM": "MMM y G",
      "GyMMMd": "d בMMM y G",
      "GyMMMED": "E, d בMMM y G",
      "h": "h a",
      "H": "H",
      "hm": "h:mm a",
      "Hm": "H:mm",
      "hms": "h:mm:ss a",
      "Hms": "H:mm:ss",
      "hmsv": "h:mm:ss a v",
      "Hmsv": "HH:mm:ss v",
      "hmv": "h:mm a v",
      "Hmv": "HH:mm v",
      "M": "L",
      "Md": "d.M",
      "MEd": "E, d.M",
      "MMM": "LLL",
      "MMMd": "d בMMM",
      "MMMED": "E, d בMMM",
      "MMMMd": "d בMMMM",
      "MMMMW-count-one": "שבוע W בMMM",
      "MMMMW-count-two": "שבוע W בMMM",
      "MMMMW-count-many": "שבוע W בMMM",
      "MMMMW-count-other": "שבוע W בMMM",
      "ms": "mm:ss",
      "y": "y",
      "yM": "M.y",
      "yMd": "d.M.y",
      "yMEd": "E, d.M.y",
      "yMM": "M.y",
      "yMMM": "MMM y",
      "yMMMd": "d בMMM y",
      "yMMMED": "E, d בMMM y",
      "yMMMM": "MMMM y",
      "yQQQ": "QQQ y",

```



```

        "yQQQQ": "QQQQ y",
        "yw-count-one": "בשנת w שבוטע y",
        "yw-count-two": "y בשנת w שבוטע",
        "yw-count-many": "y בשנת w שבוטע",
        "yw-count-other": "y בשנת w שבוטע"
    },
    "appendItems": {
        "Day": "{0} ({2}: {1})",
        "Day-Of-Week": "{0} {1}",
        "Era": "{1} {0}",
        "Hour": "{0} ({2}: {1})",
        "Minute": "{0} ({2}: {1})",
        "Month": "{0} ({2}: {1})",
        "Quarter": "{0} ({2}: {1})",
        "Second": "{0} ({2}: {1})",
        "Timezone": "{0} {1}",
        "Week": "{0} ({2}: {1})",
        "Year": "{1} {0}"
    },
    "intervalFormats": {
        "intervalFormatFallback": "{0} - {1}",
        "d": {
            "d": "d-d"
        },
        "h": {
            "a": "h a - h a",
            "h": "h-h a"
        },
        "H": {
            "H": "H-H"
        },
        "hm": {
            "a": "h:mm a - h:mm a",
            "h": "h:mm-h:mm a",
            "m": "h:mm-h:mm a"
        },
        "Hm": {
            "H": "H:mm-H:mm",
            "m": "H:mm-H:mm"
        },
        "hmv": {
            "a": "h:mm a - h:mm a v",
            "h": "h:mm-h:mm a v",
            "m": "h:mm-h:mm a v"
        },
        "Hmv": {
            "H": "H:mm-H:mm v",
            "m": "H:mm-H:mm v"
        },
        "hv": {
            "a": "h a - h a v",
            "h": "h-h a v"
        },
        "Hv": {
            "H": "H-H v"
        },
        "M": {

```

```

        "M": "M-M"
    },
    "Md": {
        "d": "d.M-d.M",
        "M": "d.M-d.M"
    },
    "MED": {
        "d": "EEEE d.M-EEEE d.M",
        "M": "EEEE d.M - EEEE d.M"
    },
    "MMM": {
        "M": "MMM-MMM"
    },
    "MMMd": {
        "d": "d-d ໓MMM",
        "M": "d ໓MMM-d ໓MMM"
    },
    "MMMED": {
        "d": "EEEE, d ໓MMM - EEEE, d ໓MMM",
        "M": "EEEE, d ໓MMM - EEEE, d ໓MMM"
    },
    "MMMM": {
        "M": "LLLL-LLLL"
    },
    "Y": {
        "Y": "Y-Y"
    },
    "YM": {
        "M": "M.Y-M.Y",
        "Y": "M.Y-M.Y"
    },
    "YMd": {
        "d": "dd.M.y - dd.M.y",
        "M": "d.M.y - d.M.y",
        "Y": "d.M.y - d.M.y"
    },
    "YMED": {
        "d": "EEEE d.M.y - EEEE d.M.y",
        "M": "EEEE d.M.y - EEEE d.M.y",
        "Y": "EEEE d.M.y - EEEE d.M.y"
    },
    "YMMM": {
        "M": "MMM-MMM Y",
        "Y": "MMM Y - MMM Y"
    },
    "YMMMd": {
        "d": "d-d ໓MMM Y",
        "M": "d MMM - d MMM Y",
        "Y": "d MMM Y - d MMM Y"
    },
    "YMMMED": {
        "d": "EEEE d MMM - EEEE d MMM Y",
        "M": "EEEE d MMM - EEEE d MMM Y",
        "Y": "EEEE d MMM Y - EEEE d MMM Y"
    },
    "YMMMM": {
        "M": "MMMM-MMMM Y",

```

```
"Y": "MMMM Y-MMMM Y"
```

CA-GREGORIAN.JSX

```
{
  "main";
  {
    "he";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 13259 $",
          "_cldrVersion";
          "31.0.1";
        }
        "language";
        "he";
      }
      "dates";
      {
        "calendars";
        {
          "gregorian";
          {
            "months";
            {
              "format";
              {
                "abbreviated";
                {
                  "1";
                  "ינוני",
                  "2";
                  "פבר",
                  "3";
                  "מרץ",
                  "4";
                  "אפר",
                  "5";
                  "מאי",
                  "6";
                  "יוני",
                  "7";
                  "יולי",

```

```

        "8";
        "אוג",
        "9";
        "ספט",
        "10";
        "אוק",
        "11";
        "נוב",
        "12";
        "דצמ";
    }
    "narrow";
    {
        "1";
        "1",
        "2";
        "2",
        "3";
        "3",
        "4";
        "4",
        "5";
        "5",
        "6";
        "6",
        "7";
        "7",
        "8";
        "8",
        "9";
        "9",
        "10";
        "10",
        "11";
        "11",
        "12";
        "12";
    }
    "wide";
    {
        "1";
        "ינואר",
        "2";
        "פברואר",
        "3";
        "מרץ",
        "4";
        "אפריל",
        "5";
        "מאי",
        "6";
        "יוני",
        "7";
        "יולי",
        "8";
        "אוגוסט",
        "9";
    }

```

```

        "ספטמבר",
        "10";
        "אוקטובר",
        "11";
        "נובמבר",
        "12";
        "דצמבר";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
        "ינו'",
        "2";
        "פבר'",
        "3";
        "מרץ",
        "4";
        "אפר'",
        "5";
        "מאי",
        "6";
        "יוני",
        "7";
        "יולי",
        "8";
        "אוג'",
        "9";
        "ספט'",
        "10";
        "אוק'",
        "11";
        "נוב'",
        "12";
        "דצמ'";
    }
}
"narrow";
{
    "1";
    "1",
    "2";
    "2",
    "3";
    "3",
    "4";
    "4",
    "5";
    "5",
    "6";
    "6",
    "7";
    "7",
    "8";
    "8",
    "9";

```

```

        "9",
        "10";
    "10",
        "11";
    "11",
        "12";
    "12";
}
"wide";
{
    "1";
    "ינואר",
        "2";
    "פברואר",
        "3";
    "מרץ",
        "4";
    "אפריל",
        "5";
    "מאי",
        "6";
    "יוני",
        "7";
    "יולי",
        "8";
    "אוגוסט",
        "9";
    "ספטמבר",
        "10";
    "אוקטובר",
        "11";
    "נובמבר",
        "12";
    "דצמבר";
}
}
}
"days";
{
    "format";
    {
        "abbreviated";
        {
            "sun";
            "יום א'",
                "mon";
            "יום ב'",
                "tue";
            "יום ג'",
                "wed";
            "יום ד'",
                "thu";
            "יום ה'",
                "fri";
            "יום ו'",
                "sat";
            "שבת";
        }
    }
}

```

```

    }
    "narrow";
    {
        "sun";
        "א'",
        "mon";
        "ב'",
        "tue";
        "ג'",
        "wed";
        "ד'",
        "thu";
        "ה'",
        "fri";
        "ו'",
        "sat";
        "ש'";
    }
    "short";
    {
        "sun";
        "א'",
        "mon";
        "ב'",
        "tue";
        "ג'",
        "wed";
        "ד'",
        "thu";
        "ה'",
        "fri";
        "ו'",
        "sat";
        "ש'";
    }
    "wide";
    {
        "sun";
        "יום ראשון",
        "mon";
        "יום שני",
        "tue";
        "יום שלישי",
        "wed";
        "יום רביעי",
        "thu";
        "יום חמישי",
        "fri";
        "יום שישי",
        "sat";
        "יום שבת";
    }
    }
    "stand-alone";
    {
        "abbreviated";
        {

```

```
        "sun";
        "א' יום",
        "mon";
        "ב' יום",
        "tue";
        "ג' יום",
        "wed";
        "ד' יום",
        "thu";
        "ה' יום",
        "fri";
        "ו' יום",
        "sat";
        "שבת";
    }
    "narrow";
    {
        "sun";
        "א'",
        "mon";
        "ב'",
        "tue";
        "ג'",
        "wed";
        "ד'",
        "thu";
        "ה'",
        "fri";
        "ו'",
        "sat";
        "ש'";
    }
    "short";
    {
        "sun";
        "א'",
        "mon";
        "ב'",
        "tue";
        "ג'",
        "wed";
        "ד'",
        "thu";
        "ה'",
        "fri";
        "ו'",
        "sat";
        "ש'";
    }
    "wide";
    {
        "sun";
        "יום ראשון",
        "mon";
        "יום שני",
        "tue";
        "יום שלישי",
```



```

        "wed";
        "יום רביעי",
        "thu";
        "יום חמישי",
        "fri";
        "יום שישי",
        "sat";
        "יום שבת";
    }
}
}
"quarters";
{
    "format";
    {
        "abbreviated";
        {
            "1";
            "Q1",
            "2";
            "Q2",
            "3";
            "Q3",
            "4";
            "Q4";
        }
        "narrow";
        {
            "1";
            "1",
            "2";
            "2",
            "3";
            "3",
            "4";
            "4";
        }
        "wide";
        {
            "1";
            "1 רבעון",
            "2";
            "2 רבעון",
            "3";
            "3 רבעון",
            "4";
            "4 רבעון";
        }
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
        "Q1",
        "2";
        "Q2",

```

```

        "3";
        "Q3",
        "4";
        "Q4";
    }
    "narrow";
    {
        "1";
        "1",
        "2";
        "2",
        "3";
        "3",
        "4";
        "4";
    }
    "wide";
    {
        "1";
        "1 רבעון",
        "2";
        "2 רבעון",
        "3";
        "3 רבעון",
        "4";
        "4 רבעון";
    }
}
"dayPeriods";
{
    "format";
    {
        "abbreviated";
        {
            "midnight";
            "חצות",
            "am";
            "לפנה"צ",
            "pm";
            "אחה"צ",
            "morning1";
            "בוקר",
            "afternoon1";
            "צהריים",
            "afternoon2";
            "אחר הצהריים",
            "evening1";
            "ערב",
            "night1";
            "לילה",
            "night2";
            "לפנות בוקר";
        }
        "narrow";
        {
            "midnight";

```

```

        "חצות",
        "am";
        "לפנה"צ",
        "pm";
        "אחה"צ",
        "morning1";
        "בוקר",
        "afternoon1";
        "צהריים",
        "afternoon2";
        "אחר הצהריים",
        "evening1";
        "ערב",
        "night1";
        "לילה",
        "night2";
        "לפנות בוקר";
    }
    "wide";
    {
        "midnight";
        "חצות",
        "am";
        "לפנה"צ",
        "pm";
        "אחה"צ",
        "morning1";
        "בוקר",
        "afternoon1";
        "צהריים",
        "afternoon2";
        "אחר הצהריים",
        "evening1";
        "ערב",
        "night1";
        "לילה",
        "night2";
        "לפנות בוקר";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "midnight";
        "חצות",
        "am";
        "לפנה"צ",
        "pm";
        "אחה"צ",
        "morning1";
        "בוקר",
        "afternoon1";
        "צהריים",
        "afternoon2";
        "אחר הצהריים",
        "evening1";
    }
}

```

```

        "ערב",
        "night1";
        "לילה",
        "night2";
        "לפנות בוקר";
    }
    "narrow";
    {
        "midnight";
        "חצות",
        "am";
        "לפנה"צ",
        "pm";
        "אחה"צ",
        "morning1";
        "בוקר",
        "afternoon1";
        "צהריים",
        "afternoon2";
        "אחר הצהריים",
        "evening1";
        "ערב",
        "night1";
        "לילה",
        "night2";
        "לפנות בוקר";
    }
    "wide";
    {
        "midnight";
        "חצות",
        "am";
        "לפנה"צ",
        "pm";
        "אחה"צ",
        "morning1";
        "בוקר",
        "afternoon1";
        "צהריים",
        "afternoon2";
        "אחר הצהריים",
        "evening1";
        "ערב",
        "night1";
        "לילה",
        "night2";
        "לפנות בוקר";
    }
    }
    "eras";
    {
        "eraNames";
        {
            "0";
            "לפני הספירה",
            "0-alt-variant";
        }
    }

```

```

        "לפנה"ס",
        "1";
        "לספירה",
        "1-alt-variant";
        "CE";
    }
    "eraAbbr";
    {
        "0";
        "לפנה"ס",
        "0-alt-variant";
        "BCE",
        "1";
        "לספירה",
        "1-alt-variant";
        "CE";
    }
    "eraNarrow";
    {
        "0";
        "לפנה"ס",
        "0-alt-variant";
        "BCE",
        "1";
        "לספירה",
        "1-alt-variant";
        "CE";
    }
}
"dateFormats";
{
    "full";
    "EEEE, d מMMMM y",
    "long";
    "d מMMMM y",
    "medium";
    "d מMMM y",
    "short";
    "d.M.y";
}
"timeFormats";
{
    "full";
    "H:mm:ss zzzz",
    "long";
    "H:mm:ss z",
    "medium";
    "H:mm:ss",
    "short";
    "H:mm";
}
"dateTimeFormats";
{
    "full";
    "{1} בשעה {0}",
    "long";
    "{1} בשעה {0}",

```

```

        "medium";
        "{1}, {0}",
        "short";
        "{1}, {0}",
        "availableFormats";
    {
        "d";
        "d",
            "E";
        "ccc",
            "Ed";
        "E n-d",
            "Ehm";
        "E h:mm a",
            "EHm";
        "E H:mm",
            "Ehms";
        "E h:mm:ss a",
            "EHms";
        "E H:mm:ss",
            "Gy";
        "y G",
            "GyMMM";
        "MMM y G",
            "GyMMMd";
        "d lMMM y G",
            "GyMMMED";
        "E, d lMMM y G",
            "h";
        "h a",
            "H";
        "H",
            "hm";
        "h:mm a",
            "Hm";
        "H:mm",
            "hms";
        "h:mm:ss a",
            "Hms";
        "H:mm:ss",
            "hmsv";
        "h:mm:ss a v",
            "Hmsv";
        "HH:mm:ss v",
            "hmv";
        "h:mm a v",
            "Hmv";
        "HH:mm v",
            "M";
        "L",
            "Md";
        "d.M",
            "MED";
        "E, d.M",
            "MMM";
        "LLL",
            "MMMd";
    }

```

```

        "d ׁMMM",
        "MMMEd";
        "E, d ׁMMM",
        "MMMMd";
        "d ׁMMMM",
        "MMMMW-count-one";
        "שבוע W ׁMMM",
        "MMMMW-count-two";
        "שבוע W ׁMMM",
        "MMMMW-count-many";
        "שבוע W ׁMMM",
        "MMMMW-count-other";
        "שבוע W ׁMMMM",
        "ms";
        "mm:ss",
        "y";
        "y",
        "yM";
        "M.y",
        "yMd";
        "d.M.y",
        "yMEd";
        "E, d.M.y",
        "yMM";
        "M.y",
        "yMMM";
        "MMM y",
        "yMMMd";
        "d ׁMMM y",
        "yMMMEd";
        "E, d ׁMMM y",
        "yMMMM";
        "MMMM y",
        "yQQQ";
        "QQQ y",
        "yQQQQ";
        "QQQQ y",
        "yw-count-one";
        "שבוע w בשנת y",
        "yw-count-two";
        "שבוע w בשנת y",
        "yw-count-many";
        "שבוע w בשנת y",
        "yw-count-other";
        "שבוע w בשנת y";
    }
    "appendItems";
    {
        "Day";
        "{0} ({2}: {1})",
        "Day-Of-Week";
        "{0} {1}",
        "Era";
        "{1} {0}",
        "Hour";
        "{0} ({2}: {1})",
        "Minute";
    }

```

```

    "{0} ({2}: {1})",
    "Month";
    "{0} ({2}: {1})",
    "Quarter";
    "{0} ({2}: {1})",
    "Second";
    "{0} ({2}: {1})",
    "Timezone";
    "{0} {1}",
    "Week";
    "{0} ({2}: {1})",
    "Year";
    "{1} {0}";
  }
  "intervalFormats";
  {
    "intervalFormatFallback";
    "{0} - {1}",
    "d";
    {
      "d";
      "d-d";
    }
    "h";
    {
      "a";
      "h a - h a",
      "h";
      "h-h a";
    }
    "H";
    {
      "H";
      "H-H";
    }
    "hm";
    {
      "a";
      "h:mm a - h:mm a",
      "h";
      "h:mm-h:mm a",
      "m";
      "h:mm-h:mm a";
    }
    "Hm";
    {
      "H";
      "H:mm-H:mm",
      "m";
      "H:mm-H:mm";
    }
    "hmv";
    {
      "a";
      "h:mm a - h:mm a v",
      "h";
      "h:mm-h:mm a v",

```



```

        "m";
        "h:mm-h:mm a v";
    }
    "Hmv";
    {
        "H";
        "H:mm-H:mm v",
        "m";
        "H:mm-H:mm v";
    }
    "hv";
    {
        "a";
        "h a - h a v",
        "h";
        "h-h a v";
    }
    "Hv";
    {
        "H";
        "H-H v";
    }
    "M";
    {
        "M";
        "M-M";
    }
    "Md";
    {
        "d";
        "d.M-d.M",
        "M";
        "d.M-d.M";
    }
    "MEd";
    {
        "d";
        "EEEE d.M-EEEE d.M",
        "M";
        "EEEE d.M - EEEE d.M";
    }
    "MMM";
    {
        "M";
        "MMM-MMM";
    }
    "MMMd";
    {
        "d";
        "d-d ׀MMM",
        "M";
        "d ׀MMM-d ׀MMM";
    }
    "MMMEd";
    {
        "d";
        "EEEE, d ׀MMM - EEEE, d ׀MMM",

```

```

        "M";
        "EEEE, d MMM - EEEE, d MMM";
    }
    "MMMM";
    {
        "M";
        "LLLL-LLLL";
    }
    "Y";
    {
        "Y";
        "Y-Y";
    }
    "yM";
    {
        "M";
        "M.Y-M.Y",
        "Y";
        "M.Y-M.Y";
    }
    "yMd";
    {
        "d";
        "dd.M.Y - dd.M.Y",
        "M";
        "d.M.Y - d.M.Y",
        "Y";
        "d.M.Y - d.M.Y";
    }
    "yMEd";
    {
        "d";
        "EEEE d.M.Y - EEEE d.M.Y",
        "M";
        "EEEE d.M.Y - EEEE d.M.Y",
        "Y";
        "EEEE d.M.Y - EEEE d.M.Y";
    }
    "yMMM";
    {
        "M";
        "MMM-MMM Y",
        "Y";
        "MMM Y - MMM Y";
    }
    "yMMMd";
    {
        "d";
        "d-d MMM Y",
        "M";
        "d MMM - d MMM Y",
        "Y";
        "d MMM Y - d MMM Y";
    }
    "yMMMEd";
    {
        "d";

```

```

    "EEEE d MMM - EEEE d MMM y",
        "M";
    "EEEE d MMM - EEEE d MMM y",
        "Y";
    "EEEE d MMM Y - EEEE d MMM y";
}
"yMMMM";
{
    "M";
    "MMMM-MMMM y",
        "Y";
    "MMMM Y-MMMM Y";
}
}
}
}
}
}
}
}
}
```

INDEX.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// Load the L10n, loadCldr from ej2-base
import { loadCldr, L10n } from '@syncfusion/ej2-base';
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
// load the CLDR data files.
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as timeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames);
L10n.load({
  'he': {
    'datepicker': { placeholder: 'הזן תאריך',
      today: 'היום'
    }
  }
});
// import the datepickercomponent
class App extends React.Component {
  render() {
    return <DatePickerComponent id="datepicker" locale='he'
enableRtl={true}/>;
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
```

```
// Load the L10n, loadCldr from ej2-base
import { loadCldr, L10n } from '@syncfusion/ej2-base';
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
// load the CLDR data files.
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as timeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames);
L10n.load({
    'he': {
        'datepicker': { placeholder: 'הזן תאריך',
            today: 'היום'
        }
    }
});
// import the datepickercomponent
class App extends React.Component<{}>, {}> {
    render() {
        return <DatePickerComponent id="datepicker" locale='he'
enableRtl={true} />;
    }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

NUMBERINGSYSTEMS.JSON

```
{
  "supplemental": {
    "version": {
      "_number": "$Revision: 12732 $",
      "_unicodeVersion": "9.0.0",
      "_cldrVersion": "31"
    },
    "numberingSystems": {
      "adlm": {
        "_digits": "ᲠᲡᲢᲣᲤᲥᲦᲧᲨᲩᲪᲫᲬᲭᲮᲯᲰᲱᲲᲳᲴᲵᲶᲷᲸᲹᲺ᲻᲼ᲽᲾᲿ",
        "_type": "numeric"
      },
      "ahom": {
        "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍᱎᱏ᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱹᱺᱛᱟᱠᱣᱚᱰᱚᱨ",
        "_type": "numeric"
      },
      "arab": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "arabext": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "armn": {
        "_rules": "armenian-upper",
        "_type": "algorithmic"
      }
    }
  },
}
```

```
"armnlow": {
  "_rules": "armenian-lower",
  "_type": "algorithmic"
},
"bali": {
  "_digits": "ᮘᮙᮐᮕᮒᮑᮓᮔᮕᮖ",
  "_type": "numeric"
},
"beng": {
  "_digits": "০১২৩৪৫৬৭৮৯",
  "_type": "numeric"
},
"bhks": {
  "_digits": "ᱠᱡᱣᱤᱦᱧᱨᱪᱫᱷᱟᱴᱚᱨ",
  "_type": "numeric"
},
"brah": {
  "_digits": "·᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊",
  "_type": "numeric"
},
"cakm": {
  "_digits": "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
  "_type": "numeric"
},
"cham": {
  "_digits": "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
  "_type": "numeric"
},
"cyrl": {
  "_rules": "cyrillic-lower",
  "_type": "algorithmic"
},
"deva": {
  "_digits": "०१२३४५६७८९",
  "_type": "numeric"
},
"ethi": {
  "_rules": "ethiopic",
  "_type": "algorithmic"
},
"fullwide": {
  "_digits": "0 1 2 3 4 5 6 7 8 9",
  "_type": "numeric"
},
"geor": {
  "_rules": "georgian",
  "_type": "algorithmic"
},
"grek": {
  "_rules": "greek-upper",
  "_type": "algorithmic"
},
"greklow": {
  "_rules": "greek-lower",
  "_type": "algorithmic"
},
```

```

"gujr": {
  "_digits": "૦૧૨૩૪૫૬૭૮૯",
  "_type": "numeric"
},
"guru": {
  "_digits": "੦੧੨੩੪੫੬੭੮੯",
  "_type": "numeric"
},
"hanidays": {
  "_rules": "zh/SpelloutRules/spellout-numbering-days",
  "_type": "algorithmic"
},
"hanidec": {
  "_digits": "〇一二三四五六七八九",
  "_type": "numeric"
},
"hans": {
  "_rules": "zh/SpelloutRules/spellout-cardinal",
  "_type": "algorithmic"
},
"hansfin": {
  "_rules": "zh/SpelloutRules/spellout-cardinal-financial",
  "_type": "algorithmic"
},
"hant": {
  "_rules": "zh_Hant/SpelloutRules/spellout-cardinal",
  "_type": "algorithmic"
},
"hantfin": {
  "_rules": "zh_Hant/SpelloutRules/spellout-cardinal-financial",
  "_type": "algorithmic"
},
"hebr": {
  "_rules": "hebrew",
  "_type": "algorithmic"
},
"hmng": {
  "_digits": "᠐ᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨ",
  "_type": "numeric"
},
"java": {
  "_digits": "᠐ᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨ",
  "_type": "numeric"
},
"jpan": {
  "_rules": "ja/SpelloutRules/spellout-cardinal",
  "_type": "algorithmic"
},
"jpanfin": {
  "_rules": "ja/SpelloutRules/spellout-cardinal-financial",
  "_type": "algorithmic"
},
"kali": {
  "_digits": "᠐ᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨ",
  "_type": "numeric"
}

```

```
,
"khmr": {
  "_digits": "០១២៣៤៥៦៧៨៩",
  "_type": "numeric"
},
"knda": {
  "_digits": "೦೧೨೩೪೫೬೭೮೯",
  "_type": "numeric"
},
"lana": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"lanatham": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"laoo": {
  "_digits": "໐໑໒໓໔໕໖໗໘໑",
  "_type": "numeric"
},
"latn": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"lepc": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"limb": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"mathbold": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathdbl": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathmono": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathsanb": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathsans": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mlym": {
  "_digits": "൦൧൨൩൪൫൬൭൮൯",
```

```

    "_type": "numeric"
  },
  "modi": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "mong": {
    "_digits": "᠐᠑᠒᠓᠔᠕᠖᠗᠘᠐",
    "_type": "numeric"
  },
  "mroo": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "mtei": {
    "_digits": "᠐᠑ᠶ᠋ᠩᠸ᠋᠐ᠠᠨᠵᠤᠢ",
    "_type": "numeric"
  },
  "mymr": {
    "_digits": "၀၁၂၃၄၅၆၇၈",
    "_type": "numeric"
  },
  "mymrshan": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mymrtlng": {
    "_digits": "᠐ᠠᠨᠢᠵᠤᠢᠨᠠᠨᠢᠵᠤᠢ",
    "_type": "numeric"
  },
  "newa": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "nkoo": {
    "_digits": "᠐ᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠ",
    "_type": "numeric"
  },
  "olck": {
    "_digits": "᠐ᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠ",
    "_type": "numeric"
  },
  "orya": {
    "_digits": "୦୧୨୩୪୫୬୭୮୯",
    "_type": "numeric"
  },
  "osma": {
    "_digits": "᠐᠑ᠶ᠋ᠩᠸ᠋᠐ᠠᠨᠵᠤᠢ",
    "_type": "numeric"
  },
  "roman": {
    "_rules": "roman-upper",
    "_type": "algorithmic"
  }

```



```

    },
    "romanlow": {
      "_rules": "roman-lower",
      "_type": "algorithmic"
    },
    "saur": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    "shrd": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    "sind": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    "sinh": {
      "_digits": "ඒ෧෨෩෪෫෬෭෮෯",
      "_type": "numeric"
    },
    "sora": {
      "_digits": "0෦123456789",
      "_type": "numeric"
    },
    "sund": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    "takr": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    "talv": {
      "_digits": "0123456789",
      "_type": "numeric"
    },
    "taml": {
      "_rules": "tamil",
      "_type": "algorithmic"
    },
    "tamldec": {
      "_digits": "0௧௨௩௪௫௬௭௮௯",
      "_type": "numeric"
    },
    "telu": {
      "_digits": "0౧౨౩౪౫౬౭౮౯",
      "_type": "numeric"
    },
    "thai": {
      "_digits": "๐๑๒๓๔๕๖๗๘๙",
      "_type": "numeric"
    },
    "tibb": {
      "_digits": "༠༡༢༣༤༥༦༧༨༩",

```

```

        "_type": "numeric"
      },
      "tirh": {
        "_digits": "□□□□□□□□□□",
        "_type": "numeric"
      },
      "vaih": {
        "_digits": "ᲙᲚᲛᲜᲝᲞᲟᲠᲡᲢᲣᲤᲥᲦᲧᲨᲩᲪᲫᲬᲭᲮᲯᲰᲱᲲᲳᲴᲵᲶᲷᲸᲹᲺ᲻᲼ᲽᲾᲿ",
        "_type": "numeric"
      },
      "wara": {
        "_digits": "□□□□□□□□□□",
        "_type": "numeric"
      }
    }
  }
}

```

NUMBERINGSYSTEMS.JSX

```

{
  "supplemental";
  {
    "version";
    {
      "_number";
      "$Revision: 12732 $",
      "_unicodeVersion";
      "9.0.0",
      "_cldrVersion";
      "31";
    }
    "numberingSystems";
    {
      "adlm";
      {
        "_digits";
        "ᲠᲡᲢᲣᲤᲦᲧᲨᲩᲪᲫᲬᲭᲮᲯᲰᲱᲲᲳᲴᲵᲶᲷᲸᲹᲺ᲻᲼ᲽᲾᲿ",
        "_type";
        "numeric";
      }
      "ahom";
      {
        "_digits";
        "ᩌᩍᩎᩏᩐᩑᩒᩓᩔᩕᩖᩗᩘᩙᩚᩛᩜᩝᩞ᩟᩠ᩡᩢᩣᩤᩥᩦᩧᩨᩩᩪᩫᩬᩭᩮᩯᩰᩱᩲᩳᩴ᩵᩶᩷᩸᩹᩺᩻᩼᩽᩾᩿",
        "_type";
        "numeric";
      }
      "arab";
      {
        "_digits";
        "٠١٢٣٤٥٦٧٨٩",
        "_type";
        "numeric";
      }
      "arabext";
    }
  }
}

```

```

    {
      "_digits";
      "․۱۲۳۴۵۶۷۸۹",
      "_type";
      "numeric";
    }
    "armn";
    {
      "_rules";
      "armenian-upper",
      "_type";
      "algorithmic";
    }
    "armnlow";
    {
      "_rules";
      "armenian-lower",
      "_type";
      "algorithmic";
    }
    "bali";
    {
      "_digits";
      "ᮘᮙᮚᮛᮜᮝᮞᮟ",
      "_type";
      "numeric";
    }
    "beng";
    {
      "_digits";
      "০১২৩৪৫৬৭৮৯",
      "_type";
      "numeric";
    }
    "bhks";
    {
      "_digits";
      "ᲠᲡᲢᲣᲤᲥᲦᲧ",
      "_type";
      "numeric";
    }
    "brah";
    {
      "_digits";
      "․ᳵᳶ᳷᳸᳹ᳺ᳻᳼᳽",
      "_type";
      "numeric";
    }
    "cakm";
    {
      "_digits";
      "ᨀᨁᨂᨃᨄᨅᨆᨇᨈ",
      "_type";
      "numeric";
    }
    "cham";

```

```

    {
      "_digits";
      "□□□□□□□□□□",
      "_type";
      "numeric";
    }
    "cyrl";
    {
      "_rules";
      "cyrillic-lower",
      "_type";
      "algorithmic";
    }
    "deva";
    {
      "_digits";
      "ॐ१२३४५६७८९",
      "_type";
      "numeric";
    }
    "ethi";
    {
      "_rules";
      "ethiopic",
      "_type";
      "algorithmic";
    }
    "fullwide";
    {
      "_digits";
      "0 1 2 3 4 5 6 7 8 9",
      "_type";
      "numeric";
    }
    "geor";
    {
      "_rules";
      "georgian",
      "_type";
      "algorithmic";
    }
    "grek";
    {
      "_rules";
      "greek-upper",
      "_type";
      "algorithmic";
    }
    "greklow";
    {
      "_rules";
      "greek-lower",
      "_type";
      "algorithmic";
    }
    "gujr";

```

```

    {
      "_digits";
      "୦୧୨୩୪୫୬୭୮୯",
      "_type";
      "numeric";
    }
    "guru";
    {
      "_digits";
      "୦୧୨୩୪୫୬୭୮୯",
      "_type";
      "numeric";
    }
    "hanidays";
    {
      "_rules";
      "zh/SpelloutRules/spellout-numbering-days",
      "_type";
      "algorithmic";
    }
    "hanidec";
    {
      "_digits";
      "〇一二三四五六七八九",
      "_type";
      "numeric";
    }
    "hans";
    {
      "_rules";
      "zh/SpelloutRules/spellout-cardinal",
      "_type";
      "algorithmic";
    }
    "hansfin";
    {
      "_rules";
      "zh/SpelloutRules/spellout-cardinal-financial",
      "_type";
      "algorithmic";
    }
    "hant";
    {
      "_rules";
      "zh_Hant/SpelloutRules/spellout-cardinal",
      "_type";
      "algorithmic";
    }
    "hantfin";
    {
      "_rules";
      "zh_Hant/SpelloutRules/spellout-cardinal-financial",
      "_type";
      "algorithmic";
    }
    "hebr";

```

```

    {
      "_rules";
      "hebrew",
      "_type";
      "algorithmic";
    }
    "hmng";
    {
      "_digits";
      "□□□□□□□□□□",
      "_type";
      "numeric";
    }
    "java";
    {
      "_digits";
      "௦௧௨௩௪௫௬௭௮௯௦",
      "_type";
      "numeric";
    }
    "jpan";
    {
      "_rules";
      "ja/SpelloutRules/spellout-cardinal",
      "_type";
      "algorithmic";
    }
    "jpanfin";
    {
      "_rules";
      "ja/SpelloutRules/spellout-cardinal-financial",
      "_type";
      "algorithmic";
    }
    "kali";
    {
      "_digits";
      "□□□□□□□□□□",
      "_type";
      "numeric";
    }
    "khmr";
    {
      "_digits";
      "០១២៣៤៥៦៧៨៩",
      "_type";
      "numeric";
    }
    "knda";
    {
      "_digits";
      "೦೧೨೩೪೫೬೭೮೯",
      "_type";
      "numeric";
    }
  }

```

```
"lana";
{
  "_digits";
  "□□□□□□□□□□",
  "_type";
  "numeric";
}
"lanatham";
{
  "_digits";
  "□□□□□□□□□□",
  "_type";
  "numeric";
}
"laoo";
{
  "_digits";
  "໐໑໒໓໔໕໖໗໘໑",
  "_type";
  "numeric";
}
"latn";
{
  "_digits";
  "0123456789",
  "_type";
  "numeric";
}
"lepc";
{
  "_digits";
  "□□□□□□□□□□",
  "_type";
  "numeric";
}
"limb";
{
  "_digits";
  "□□□□□□□□□□",
  "_type";
  "numeric";
}
"mathbold";
{
  "_digits";
  "0123456789",
  "_type";
  "numeric";
}
"mathdbl";
{
  "_digits";
  "0123456789",
  "_type";
  "numeric";
}
```

```
"mathmono";
{
    "_digits";
    "0123456789",
    "_type";
    "numeric";
}
"mathsansb";
{
    "_digits";
    "0123456789",
    "_type";
    "numeric";
}
"mathsans";
{
    "_digits";
    "0123456789",
    "_type";
    "numeric";
}
"mlym";
{
    "_digits";
    "൦൧൨൩൪൫൬൭൮൯",
    "_type";
    "numeric";
}
"modi";
{
    "_digits";
    "□□□□□□□□",
    "_type";
    "numeric";
}
"mong";
{
    "_digits";
    "᠐᠑᠒᠓᠐ᠠᠨᠨᠠᠭ",
    "_type";
    "numeric";
}
"mroo";
{
    "_digits";
    "□□□□□□□□",
    "_type";
    "numeric";
}
"mtei";
{
    "_digits";
    "ႤႤႬႬႭႭႬႬႬႬ",
    "_type";
    "numeric";
}
```



```
"mymr";
{
  "_digits";
  "၀၁၂၃၄၅၆၇၈၉",
  "_type";
  "numeric";
}
"mymrshan";
{
  "_digits";
  "၀၁၂၃၄၅၆၇၈၉",
  "_type";
  "numeric";
}
"mymrtlng";
{
  "_digits";
  "၀၁၂၃၄၅၆၇၈၉",
  "_type";
  "numeric";
}
"newa";
{
  "_digits";
  "၀၁၂၃၄၅၆၇၈၉",
  "_type";
  "numeric";
}
"nkoo";
{
  "_digits";
  "၀၁၂၃၄၅၆၇၈၉",
  "_type";
  "numeric";
}
"olck";
{
  "_digits";
  "၀၁၂၃၄၅၆၇၈၉",
  "_type";
  "numeric";
}
"orya";
{
  "_digits";
  "၀၁၂၃၄၅၆၇၈၉",
  "_type";
  "numeric";
}
"osma";
{
  "_digits";
  "၀၁၂၃၄၅၆၇၈၉",
  "_type";
  "numeric";
}
```

```

        "numeric";
    }
    "roman";
    {
        "_rules";
        "roman-upper",
        "_type";
        "algorithmic";
    }
    "romanlow";
    {
        "_rules";
        "roman-lower",
        "_type";
        "algorithmic";
    }
    "saur";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "shrd";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "sind";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "sinh";
    {
        "_digits";
        "⁂௮௮௮௮௮௮௮௮",
        "_type";
        "numeric";
    }
    "sora";
    {
        "_digits";
        "0௧23456789",
        "_type";
        "numeric";
    }
    "sund";
    {
        "_digits";
        "□□□□□□□□",
        "_type";

```

```

        "numeric";
    }
    "takr";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
    "tal";
    {
        "_digits";
        "ᲠᲣᲚᲘᲗᲣᲚᲘᲗ",
        "_type";
        "numeric";
    }
    "taml";
    {
        "_rules";
        "tamil",
        "_type";
        "algorithmic";
    }
    "tamldec";
    {
        "_digits";
        "௦௧௨௩௪௫௬௭௮௯",
        "_type";
        "numeric";
    }
    "telu";
    {
        "_digits";
        "౦౧౨౩౪౫౬౭౮౯",
        "_type";
        "numeric";
    }
    "thai";
    {
        "_digits";
        "๐๑๒๓๔๕๖๗๘๙",
        "_type";
        "numeric";
    }
    "tib";
    {
        "_digits";
        "༠༡༢༣༤༥༦༧༨༩",
        "_type";
        "numeric";
    }
    "tirh";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";

```

```

        "numeric";
    }
    "vaii";
    {
        "_digits";
        "ᲘᲙᲚᲛᲞᲟᲠᲡᲢᲣᲤᲥᲦᲧᲨᲩᲪᲫᲬᲭᲮᲯᲰᲱᲲᲳᲴᲵᲶᲷᲸᲹᲺ᲻᲼ᲽᲾᲿ",
        "_type";
        "numeric";
    }
    "wara";
    {
        "_digits";
        "ᲚᲛᲞᲟᲠᲡᲢᲣᲤᲥᲦᲧᲨᲩᲪᲫᲬᲭᲮᲯᲰᲱᲲᲳᲴᲵᲶᲷᲸᲹᲺ᲻᲼ᲽᲾᲿ",
        "_type";
        "numeric";
    }
}

```

NUMBERS.JSON

```

{
  "main": {
    "he": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31.0.1"
        },
        "language": "he"
      },
      "numbers": {
        "defaultNumberingSystem": "latn",
        "otherNumberingSystems": {
          "native": "latn",
          "traditional": "hebr"
        },
        "minimumGroupingDigits": "1",
        "symbols-numberSystem-latn": {
          "decimal": ".",
          "group": ",",
          "list": ";",
          "percentSign": "%",
          "plusSign": "+",
          "minusSign": "-",
          "exponential": "E",
          "superscriptingExponent": "x",
          "perMille": "‰",
          "infinity": "∞",
          "nan": "NaN",
          "timeSeparator": ":"
        },
        "decimalFormats-numberSystem-latn": {
          "standard": "#,##0.###",
          "long": {

```

```

    "decimalFormat": {
      "1000-count-one": "אלף 0",
      "1000-count-two": "אלף 0",
      "1000-count-many": "אלף 0",
      "1000-count-other": "אלף 0",
      "10000-count-one": "אלף 00",
      "10000-count-two": "אלף 00",
      "10000-count-many": "אלף 00",
      "10000-count-other": "אלף 00",
      "100000-count-one": "אלף 000",
      "100000-count-two": "אלף 000",
      "100000-count-many": "אלף 000",
      "100000-count-other": "אלף 000",
      "1000000-count-one": "0 מיליון",
      "1000000-count-two": "0 מיליון",
      "1000000-count-many": "0 מיליון",
      "1000000-count-other": "0 מיליון",
      "10000000-count-one": "00 מיליון",
      "10000000-count-two": "00 מיליון",
      "10000000-count-many": "00 מיליון",
      "10000000-count-other": "00 מיליון",
      "100000000-count-one": "000 מיליון",
      "100000000-count-two": "000 מיליון",
      "100000000-count-many": "000 מיליון",
      "100000000-count-other": "000 מיליון",
      "1000000000-count-one": "0 מיליארד",
      "1000000000-count-two": "0 מיליארד",
      "1000000000-count-many": "0 מיליארד",
      "1000000000-count-other": "0 מיליארד",
      "10000000000-count-one": "00 מיליארד",
      "10000000000-count-two": "00 מיליארד",
      "10000000000-count-many": "00 מיליארד",
      "10000000000-count-other": "00 מיליארד",
      "100000000000-count-one": "000 מיליארד",
      "100000000000-count-two": "000 מיליארד",
      "100000000000-count-many": "000 מיליארד",
      "100000000000-count-other": "000 מיליארד",
      "1000000000000-count-one": "0 טריליון",
      "1000000000000-count-two": "0 טריליון",
      "1000000000000-count-many": "0 טריליון",
      "1000000000000-count-other": "0 טריליון",
      "10000000000000-count-one": "00 טריליון",
      "10000000000000-count-two": "00 טריליון",
      "10000000000000-count-many": "00 טריליון",
      "10000000000000-count-other": "00 טריליון",
      "100000000000000-count-one": "000 טריליון",
      "100000000000000-count-two": "000 טריליון",
      "100000000000000-count-many": "000 טריליון",
      "100000000000000-count-other": "000 טריליון"
    }
  },
  "short": {
    "decimalFormat": {
      "1000-count-one": "0K",
      "1000-count-two": "0K",
      "1000-count-many": "0K",
      "1000-count-other": "0K",

```

```

        "10000-count-one": "00K",
        "10000-count-two": "00K",
        "10000-count-many": "00K",
        "10000-count-other": "00K",
        "100000-count-one": "000K",
        "100000-count-two": "000K",
        "100000-count-many": "000K",
        "100000-count-other": "000K",
        "1000000-count-one": "0M",
        "1000000-count-two": "0M",
        "1000000-count-many": "0M",
        "1000000-count-other": "0M",
        "10000000-count-one": "00M",
        "10000000-count-two": "00M",
        "10000000-count-many": "00M",
        "10000000-count-other": "00M",
        "100000000-count-one": "000M",
        "100000000-count-two": "000M",
        "100000000-count-many": "000M",
        "100000000-count-other": "000M",
        "1000000000-count-one": "0B",
        "1000000000-count-two": "0B",
        "1000000000-count-many": "0B",
        "1000000000-count-other": "0B",
        "10000000000-count-one": "00B",
        "10000000000-count-two": "00B",
        "10000000000-count-many": "00B",
        "10000000000-count-other": "00B",
        "100000000000-count-one": "000B",
        "100000000000-count-two": "000B",
        "100000000000-count-many": "000B",
        "100000000000-count-other": "000B",
        "1000000000000-count-one": "0T",
        "1000000000000-count-two": "0T",
        "1000000000000-count-many": "0T",
        "1000000000000-count-other": "0T",
        "10000000000000-count-one": "00T",
        "10000000000000-count-two": "00T",
        "10000000000000-count-many": "00T",
        "10000000000000-count-other": "00T",
        "100000000000000-count-one": "000T",
        "100000000000000-count-two": "000T",
        "100000000000000-count-many": "000T",
        "100000000000000-count-other": "000T"
    }
}
},
"scientificFormats-numberSystem-latn": {
    "standard": "#E0"
},
"percentFormats-numberSystem-latn": {
    "standard": "#,##0%"
},
"currencyFormats-numberSystem-latn": {
    "currencySpacing": {
        "beforeCurrency": {
            "currencyMatch": "[:^S:]",

```

```

    "surroundingMatch": "[:digit:]",
    "insertBetween": " "
  },
  "afterCurrency": {
    "currencyMatch": "[:^S:]",
    "surroundingMatch": "[:digit:]",
    "insertBetween": " "
  }
},
"standard": "##0.00#, -¤; ##0.00#, ¤",
"accounting": "#,##0.00 ¤",
"short": {
  "standard": {
    "1000-count-one": "¤ 0K",
    "1000-count-two": "¤ 0K",
    "1000-count-many": "¤0K",
    "1000-count-other": "¤ 0K",
    "10000-count-one": "¤00K",
    "10000-count-two": "¤00K",
    "10000-count-many": "¤00K",
    "10000-count-other": "¤ 00K",
    "100000-count-one": "¤000K",
    "100000-count-two": "¤000K",
    "100000-count-many": "¤000K",
    "100000-count-other": "¤000K",
    "1000000-count-one": "¤0M",
    "1000000-count-two": "¤0M",
    "1000000-count-many": "¤0M",
    "1000000-count-other": "¤0M",
    "10000000-count-one": "¤00M",
    "10000000-count-two": "¤00M",
    "10000000-count-many": "¤00M",
    "10000000-count-other": "¤00M",
    "100000000-count-one": "¤000M",
    "100000000-count-two": "¤000M",
    "100000000-count-many": "¤000M",
    "100000000-count-other": "¤000M",
    "1000000000-count-one": "¤0B",
    "1000000000-count-two": "¤0B",
    "1000000000-count-many": "¤0B",
    "1000000000-count-other": "¤0B",
    "10000000000-count-one": "¤00B",
    "10000000000-count-two": "¤00B",
    "10000000000-count-many": "¤00B",
    "10000000000-count-other": "¤00B",
    "100000000000-count-one": "¤000B",
    "100000000000-count-two": "¤000B",
    "100000000000-count-many": "¤000B",
    "100000000000-count-other": "¤000B",
    "1000000000000-count-one": "¤0T",
    "1000000000000-count-two": "¤0T",
    "1000000000000-count-many": "¤0T",
    "1000000000000-count-other": "¤0T",
    "10000000000000-count-one": "¤00T",
    "10000000000000-count-two": "¤00T",
    "10000000000000-count-many": "¤00T",
    "10000000000000-count-other": "¤00T",

```

```

        "1000000000000000-count-one": "א000T",
        "1000000000000000-count-two": "א000T",
        "1000000000000000-count-many": "א000T",
        "1000000000000000-count-other": "א000T"
    },
    },
    "unitPattern-count-one": "{0} {1}",
    "unitPattern-count-two": "{0} {1}",
    "unitPattern-count-many": "{0} {1}",
    "unitPattern-count-other": "{0} {1}"
},
"miscPatterns-numberSystem-latn": {
    "atLeast": "≥{0}+",
    "range": "{0}-{1}"
},
"minimalPairs": {
    "pluralMinimalPairs": "שנה",
    "pluralMinimalPairs": "שנתיים",
    "pluralMinimalPairs": "{0} שנה",
    "pluralMinimalPairs": "{0} שנים",
    "other": "פנה ימינה בפנייה ה-{0}"
}
}
}
}
}
}

```

NUMBERS.JSX

```

{
    "main";
    {
        "he";
        {
            "identity";
            {
                "version";
                {
                    "_number";
                    "$Revision: 13259 $",
                    "_cldrVersion";
                    "31.0.1";
                }
                "language";
                "he";
            }
            "numbers";
            {
                "defaultNumberingSystem";
                "latn",
                "otherNumberingSystems";
                {
                    "native";
                    "latn",
                    "traditional";
                    "hebr";
                }
            }
        }
    }
}

```



```

    }
    "minimumGroupingDigits";
    "1",
      "symbols-numberSystem-latn";
    {
      "decimal";
      ".",
        "group";
      ",",
        "list";
      ";",
        "percentSign";
      "%",
        "plusSign";
      "+",
        "minusSign";
      "-",
        "exponential";
      "E",
        "superscriptingExponent";
      "x",
        "perMille";
      "‰",
        "infinity";
      "∞",
        "nan";
      "NaN",
        "timeSeparator";
      ":";
    }
    "decimalFormats-numberSystem-latn";
    {
      "standard";
      "#,##0.###",
        "long";
      {
        "decimalFormat";
        {
          "1000-count-one";
          "q7x 0",
            "1000-count-two";
          "q7x 0",
            "1000-count-many";
          "q7x 0",
            "1000-count-other";
          "q7x 0",
            "10000-count-one";
          "q7x 00",
            "10000-count-two";
          "q7x 00",
            "10000-count-many";
          "q7x 00",
            "10000-count-other";
          "q7x 00",
            "100000-count-one";
          "q7x 000",
            "100000-count-two";
        }
      }
    }
  }

```

```
"אלף 000",
    "100000-count-many";
"אלף 000",
    "100000-count-other";
"אלף 000",
    "1000000-count-one";
"מיליון 0",
    "1000000-count-two";
"מיליון 0",
    "1000000-count-many";
"מיליון 0",
    "1000000-count-other";
"מיליון 0",
    "10000000-count-one";
"מיליון 00",
    "10000000-count-two";
"מיליון 00",
    "10000000-count-many";
"מיליון 00",
    "10000000-count-other";
"מיליון 00",
    "100000000-count-one";
"מיליון 000",
    "100000000-count-two";
"מיליון 000",
    "100000000-count-many";
"מיליון 000",
    "100000000-count-other";
"מיליון 000",
    "1000000000-count-one";
"מיליארד 0",
    "1000000000-count-two";
"מיליארד 0",
    "1000000000-count-many";
"מיליארד 0",
    "1000000000-count-other";
"מיליארד 0",
    "10000000000-count-one";
"מיליארד 00",
    "10000000000-count-two";
"מיליארד 00",
    "10000000000-count-many";
"מיליארד 00",
    "10000000000-count-other";
"מיליארד 00",
    "100000000000-count-one";
"מיליארד 000",
    "100000000000-count-two";
"מיליארד 000",
    "100000000000-count-many";
"מיליארד 000",
    "100000000000-count-other";
"מיליארד 000",
    "1000000000000-count-one";
"טריליון 0",
    "1000000000000-count-two";
"טריליון 0",
```

```

        "1000000000000-count-many";
        "טרייליון 0",
        "1000000000000-count-other";
        "טרייליון 0",
        "1000000000000-count-one";
        "טרייליון 00",
        "1000000000000-count-two";
        "טרייליון 00",
        "1000000000000-count-many";
        "טרייליון 00",
        "1000000000000-count-other";
        "טרייליון 00",
        "1000000000000-count-one";
        "טרייליון 000",
        "1000000000000-count-two";
        "טרייליון 000",
        "1000000000000-count-many";
        "טרייליון 000",
        "1000000000000-count-other";
        "טרייליון 000";
    }
}
"short";
{
    "decimalFormat";
    {
        "1000-count-one";
        "0K",
        "1000-count-two";
        "0K",
        "1000-count-many";
        "0K",
        "1000-count-other";
        "0K",
        "10000-count-one";
        "00K",
        "10000-count-two";
        "00K",
        "10000-count-many";
        "00K",
        "10000-count-other";
        "00K",
        "100000-count-one";
        "000K",
        "100000-count-two";
        "000K",
        "100000-count-many";
        "000K",
        "100000-count-other";
        "000K",
        "1000000-count-one";
        "0M",
        "1000000-count-two";
        "0M",
        "1000000-count-many";
        "0M",
        "1000000-count-other";
    }
}

```

```
"0M",
    "10000000-count-one";
"00M",
    "10000000-count-two";
"00M",
    "10000000-count-many";
"00M",
    "10000000-count-other";
"00M",
    "100000000-count-one";
"000M",
    "100000000-count-two";
"000M",
    "100000000-count-many";
"000M",
    "100000000-count-other";
"000M",
    "1000000000-count-one";
"0B",
    "1000000000-count-two";
"0B",
    "1000000000-count-many";
"0B",
    "1000000000-count-other";
"0B",
    "10000000000-count-one";
"00B",
    "10000000000-count-two";
"00B",
    "10000000000-count-many";
"00B",
    "10000000000-count-other";
"00B",
    "100000000000-count-one";
"000B",
    "100000000000-count-two";
"000B",
    "100000000000-count-many";
"000B",
    "100000000000-count-other";
"000B",
    "1000000000000-count-one";
"0T",
    "1000000000000-count-two";
"0T",
    "1000000000000-count-many";
"0T",
    "1000000000000-count-other";
"0T",
    "10000000000000-count-one";
"00T",
    "10000000000000-count-two";
"00T",
    "10000000000000-count-many";
"00T",
    "10000000000000-count-other";
"00T",
    "100000000000000-count-one";
"00T",
    "100000000000000-count-other";
```

```

        "1000000000000000-count-one";
        "000T",
        "1000000000000000-count-two";
        "000T",
        "1000000000000000-count-many";
        "000T",
        "1000000000000000-count-other";
        "000T";
    }
}
}
"scientificFormats-numberSystem-latn";
{
    "standard";
    "#E0";
}
"percentFormats-numberSystem-latn";
{
    "standard";
    "#,##0%";
}
"currencyFormats-numberSystem-latn";
{
    "currencySpacing";
    {
        "beforeCurrency";
        {
            "currencyMatch";
            "[:^S:]",
            "surroundingMatch";
            "[:digit:]",
            "insertBetween";
            " ";
        }
        "afterCurrency";
        {
            "currencyMatch";
            "[:^S:]",
            "surroundingMatch";
            "[:digit:]",
            "insertBetween";
            " ";
        }
    }
}
"standard";
"##0.00#,-¤; ##0.00#, ¤",
    "accounting";
"#,##0.00 ¤",
    "short";
{
    "standard";
    {
        "1000-count-one";
        "¤ OK",
        "1000-count-two";
        "¤ OK",
        "1000-count-many";
    }
}

```

```
"K",
    "1000-count-other";
" OK",
    "10000-count-one";
"00K",
    "10000-count-two";
"00K",
    "10000-count-many";
"00K",
    "10000-count-other";
" OK",
    "100000-count-one";
"000K",
    "100000-count-two";
"000K",
    "100000-count-many";
"000K",
    "100000-count-other";
"000K",
    "1000000-count-one";
"0M",
    "1000000-count-two";
"0M",
    "1000000-count-many";
"0M",
    "1000000-count-other";
"0M",
    "10000000-count-one";
"00M",
    "10000000-count-two";
"00M",
    "10000000-count-many";
"00M",
    "10000000-count-other";
"00M",
    "100000000-count-one";
"000M",
    "100000000-count-two";
"000M",
    "100000000-count-many";
"000M",
    "100000000-count-other";
"000M",
    "1000000000-count-one";
"0B",
    "1000000000-count-two";
"0B",
    "1000000000-count-many";
"0B",
    "1000000000-count-other";
"0B",
    "10000000000-count-one";
"00B",
    "10000000000-count-two";
"00B",
    "10000000000-count-many";
"00B",
```

```

        "10000000000-count-other";
        "¤00B",
        "100000000000-count-one";
        "¤000B",
        "1000000000000-count-two";
        "¤000B",
        "10000000000000-count-many";
        "¤000B",
        "100000000000000-count-other";
        "¤000B",
        "1000000000000000-count-one";
        "¤0T",
        "1000000000000000-count-two";
        "¤0T",
        "10000000000000000-count-many";
        "¤0T",
        "100000000000000000-count-other";
        "¤0T",
        "1000000000000000000-count-one";
        "¤00T",
        "1000000000000000000-count-two";
        "¤00T",
        "10000000000000000000-count-many";
        "¤00T",
        "100000000000000000000-count-other";
        "¤00T",
        "1000000000000000000000-count-one";
        "¤000T",
        "10000000000000000000000-count-two";
        "¤000T",
        "100000000000000000000000-count-many";
        "¤000T",
        "1000000000000000000000000-count-other";
        "¤000T";
    }
}
"unitPattern-count-one";
"{0} {1}",
    "unitPattern-count-two";
"{0} {1}",
    "unitPattern-count-many";
"{0} {1}",
    "unitPattern-count-other";
"{0} {1}";
}
"miscPatterns-numberSystem-latn";
{
    "atLeast";
    "≥{0}+",
        "range";
    "{0}-{1}";
}
"minimalPairs";
{
    "pluralMinimalPairs";
    "הַיּוֹם",
        "pluralMinimalPairs";

```

```

        "שנתיים",
        "pluralMinimalPairs";
    "{0} שנה",
    "pluralMinimalPairs";
    "{0} שנים",
    "other";
    "פנה ימינה בפנייה ה-{0}";
  }
}
}
}
}

```

TIMEZONENAMES.JSON

```

{
  "main": {
    "he": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31.0.1"
        },
        "language": "he"
      },
      "dates": {
        "timeZoneNames": {
          "hourFormat": "+HH:mm;-HH:mm",
          "gmtFormat": "GMT{0}",
          "gmtZeroFormat": "GMT",
          "regionFormat": "שעון {0}",
          "regionFormat-type-daylight": "קיצ {0} (שעון",
          "regionFormat-type-standard": "חורף {0} (שעון",
          "fallbackFormat": "{1} ({0})",
          "zone": {
            "America": {
              "Adak": {
                "exemplarCity": "אדאק"
              },
              "Anchorage": {
                "exemplarCity": "אנקורג'י"
              },
              "Anguilla": {
                "exemplarCity": "אנגווילה"
              },
              "Antigua": {
                "exemplarCity": "אנטיגואה"
              },
              "Araguaina": {
                "exemplarCity": "אראגואינה"
              },
              "Argentina": {
                "Rio_Gallegos": {
                  "exemplarCity": "ריו גאייגוס"
                },
                "San_Juan": {

```



```

        "exemplarCity": "סן חואן"
    },
    "Ushuaia": {
        "exemplarCity": "אושואיה"
    },
    "La_Rioja": {
        "exemplarCity": "לה ריוחה"
    },
    "San_Luis": {
        "exemplarCity": "סן לואיס"
    },
    "Salta": {
        "exemplarCity": "סלטה"
    },
    "Tucuman": {
        "exemplarCity": "טוקומן"
    }
},
"Aruba": {
    "exemplarCity": "ארובה"
},
"Asuncion": {
    "exemplarCity": "אסונסיון"
},
"Bahia": {
    "exemplarCity": "באהיה"
},
"Bahia_Banderas": {
    "exemplarCity": "באהיה בנדרס"
},
"Barbados": {
    "exemplarCity": "ברבדוס"
},
"Belem": {
    "exemplarCity": "בלם"
},
"Belize": {
    "exemplarCity": "בליז"
},
"Blanc-Sablon": {
    "exemplarCity": "בלאן-סבלון"
},
"Boa_Vista": {
    "exemplarCity": "בואה ויסטה"
},
"Bogota": {
    "exemplarCity": "בוגוטה"
},
"Boise": {
    "exemplarCity": "בויסי"
},
"Buenos_Aires": {
    "exemplarCity": "בואנוס איירס"
},
"Cambridge_Bay": {
    "exemplarCity": "קיימברידג' ביי"
},

```

```
"Campo_Grande": {
  "exemplarCity": "קמפו גרנדה"
},
"Cancun": {
  "exemplarCity": "קנקון"
},
"Caracas": {
  "exemplarCity": "קראקס"
},
"Catamarca": {
  "exemplarCity": "קטמרקה"
},
"Cayenne": {
  "exemplarCity": "קאייין"
},
"Cayman": {
  "exemplarCity": "קיימן"
},
"Chicago": {
  "exemplarCity": "שיקגו"
},
"Chihuahua": {
  "exemplarCity": "צ'יוואוואה"
},
"Coral_Harbour": {
  "exemplarCity": "אטיקוקן"
},
"Cordoba": {
  "exemplarCity": "קורדובה"
},
"Costa_Rica": {
  "exemplarCity": "קוסטה ריקה"
},
"Creston": {
  "exemplarCity": "קרטסון"
},
"Cuiaba": {
  "exemplarCity": "קויאבה"
},
"Curacao": {
  "exemplarCity": "קוראסאו"
},
"Danmarkshavn": {
  "exemplarCity": "דנמרקסהוון"
},
"Dawson": {
  "exemplarCity": "דוסון"
},
"Dawson_Creek": {
  "exemplarCity": "דוסון קריק"
},
"Denver": {
  "exemplarCity": "דנוור"
},
"Detroit": {
  "exemplarCity": "דטרויט"
},
```

```
"Dominica": {
  "exemplarCity": "דומיניקה"
},
"Edmonton": {
  "exemplarCity": "אדמונטון"
},
"Eirunepe": {
  "exemplarCity": "אירונפי"
},
"El_Salvador": {
  "exemplarCity": "אל סלבדור"
},
"Fort_Nelson": {
  "exemplarCity": "פורט נלסון"
},
"Fortaleza": {
  "exemplarCity": "פורטאלזה"
},
"Glace_Bay": {
  "exemplarCity": "גלייס ביי"
},
"Godthab": {
  "exemplarCity": "נואוק"
},
"Goose_Bay": {
  "exemplarCity": "גוס ביי"
},
"Grand_Turk": {
  "exemplarCity": "גרנד טורק"
},
"Grenada": {
  "exemplarCity": "גרנדה"
},
"Guadeloupe": {
  "exemplarCity": "גואדלופ"
},
"Guatemala": {
  "exemplarCity": "גואטמלה"
},
"Guayaquil": {
  "exemplarCity": "גואיאקיל"
},
"Guyana": {
  "exemplarCity": "גיאנה"
},
"Halifax": {
  "exemplarCity": "הליפקס"
},
"Havana": {
  "exemplarCity": "הוואנה"
},
"Hermosillo": {
  "exemplarCity": "הרמוסיו"
},
"Indiana": {
  "Vincennes": {
    "exemplarCity": "וינסנס, אינדיאנה"
  }
}
```

```

    },
    "Petersburg": {
      "exemplarCity": "פיטרסבורג, אינדיאנה"
    },
    "Tell_City": {
      "exemplarCity": "טל סיטי, אינדיאנה"
    },
    "Knox": {
      "exemplarCity": "נוקס, אינדיאנה"
    },
    "Winamac": {
      "exemplarCity": "ויןמאק, אינדיאנה"
    },
    "Marengo": {
      "exemplarCity": "מרנגו, אינדיאנה"
    },
    "Vevay": {
      "exemplarCity": "ויוואיי, אינדיאנה"
    }
  },
  "Indianapolis": {
    "exemplarCity": "אינדיאנפוליס"
  },
  "Inuvik": {
    "exemplarCity": "אינוויק"
  },
  "Iqaluit": {
    "exemplarCity": "איקלואיט"
  },
  "Jamaica": {
    "exemplarCity": "ג'מייקה"
  },
  "Jujuy": {
    "exemplarCity": "חוחוי"
  },
  "Juneau": {
    "exemplarCity": "ג'וננו"
  },
  "Kentucky": {
    "Monticello": {
      "exemplarCity": "מונטיצ'לו, קנטאקי"
    }
  },
  "Kralendijk": {
    "exemplarCity": "קרלנדייק"
  },
  "La_Paz": {
    "exemplarCity": "לה פאס"
  },
  "Lima": {
    "exemplarCity": "לימה"
  },
  "Los_Angeles": {
    "exemplarCity": "לוס אנג'לס"
  },
  "Louisville": {
    "exemplarCity": "לואיוויל"
  }

```

```
    },  
    "Lower_Princes": {  
      "exemplarCity": "לואוור פרינסס קוורטר"  
    },  
    "Maceio": {  
      "exemplarCity": "מסיאון"  
    },  
    "Managua": {  
      "exemplarCity": "מנגואה"  
    },  
    "Manaus": {  
      "exemplarCity": "מנאוס"  
    },  
    "Marigot": {  
      "exemplarCity": "מריגון"  
    },  
    "Martinique": {  
      "exemplarCity": "מרטיניק"  
    },  
    "Matamoros": {  
      "exemplarCity": "מטמורוס"  
    },  
    "Mazatlan": {  
      "exemplarCity": "מזטלן"  
    },  
    "Mendoza": {  
      "exemplarCity": "מנדוזזה"  
    },  
    "Menominee": {  
      "exemplarCity": "מנומיני"  
    },  
    "Merida": {  
      "exemplarCity": "מרידה"  
    },  
    "Metlakatla": {  
      "exemplarCity": "מטלקטלה"  
    },  
    "Mexico_City": {  
      "exemplarCity": "מקסיקו סיטי"  
    },  
    "Miquelon": {  
      "exemplarCity": "מיקלון"  
    },  
    "Moncton": {  
      "exemplarCity": "מונקטון"  
    },  
    "Monterrey": {  
      "exemplarCity": "מונטריי"  
    },  
    "Montevideo": {  
      "exemplarCity": "מונטווידאו"  
    },  
    "Montserrat": {  
      "exemplarCity": "מונטראט"  
    },  
    "Nassau": {  
      "exemplarCity": "נסאון"
```

```
    },
    "New_York": {
      "exemplarCity": "ניו יורק"
    },
    "Nipigon": {
      "exemplarCity": "ניפיגון"
    },
    "Nome": {
      "exemplarCity": "נום"
    },
    "Noronha": {
      "exemplarCity": "נורוניה"
    },
    "North_Dakota": {
      "Beulah": {
        "exemplarCity": "ביולה, צפון דקוטה"
      },
      "New_Salem": {
        "exemplarCity": "ניו סיילס, צפון דקוטה"
      },
      "Center": {
        "exemplarCity": "סנטר, צפון דקוטה"
      }
    },
    "Ojinaga": {
      "exemplarCity": "אוג'ינאגה"
    },
    "Panama": {
      "exemplarCity": "פנמה"
    },
    "Pangnirtung": {
      "exemplarCity": "פנגנירטונג"
    },
    "Paramaribo": {
      "exemplarCity": "פרמריבו"
    },
    "Phoenix": {
      "exemplarCity": "פיניקס"
    },
    "Port-au-Prince": {
      "exemplarCity": "פורט או פראנס"
    },
    "Port_of_Spain": {
      "exemplarCity": "פורט אוף ספייין"
    },
    "Porto_Velho": {
      "exemplarCity": "פורטו וליו"
    },
    "Puerto_Rico": {
      "exemplarCity": "פוארטו ריקו"
    },
    "Rainy_River": {
      "exemplarCity": "רייני ריבר"
    },
    "Rankin_Inlet": {
      "exemplarCity": "רנקין אינלט"
    },
  },
```

```
"Recife": {
  "exemplarCity": "רסיפה"
},
"Regina": {
  "exemplarCity": "רג'ינה"
},
"Resolute": {
  "exemplarCity": "רזולוט"
},
"Rio_Branco": {
  "exemplarCity": "ריו ברנקו"
},
"Santa_Isabel": {
  "exemplarCity": "סנטה איסבל"
},
"Santarem": {
  "exemplarCity": "סנטרם"
},
"Santiago": {
  "exemplarCity": "סנטיאגו"
},
"Santo_Domingo": {
  "exemplarCity": "סנטו דומינגו"
},
"Sao_Paulo": {
  "exemplarCity": "סאו פאולו"
},
"Scoresbysund": {
  "exemplarCity": "סקורסביסונד"
},
"Sitka": {
  "exemplarCity": "סיטקה"
},
"St_Barthelemy": {
  "exemplarCity": "סנט ברתלמי"
},
"St_Johns": {
  "exemplarCity": "סנט ג'ונס"
},
"St_Kitts": {
  "exemplarCity": "סנט קיטס"
},
"St_Lucia": {
  "exemplarCity": "סנט לוסיה"
},
"St_Thomas": {
  "exemplarCity": "סנט תומאס"
},
"St_Vincent": {
  "exemplarCity": "סנט וינסנט"
},
"Swift_Current": {
  "exemplarCity": "סוויפט קרנט"
},
"Tegucigalpa": {
  "exemplarCity": "טגוסיגלפה"
},
}
```

```

    "Thule": {
      "exemplarCity": "תולה"
    },
    "Thunder_Bay": {
      "exemplarCity": "ת'אנדר ביי"
    },
    "Tijuana": {
      "exemplarCity": "טיחואנה"
    },
    "Toronto": {
      "exemplarCity": "טורונטו"
    },
    "Tortola": {
      "exemplarCity": "טורטולה"
    },
    "Vancouver": {
      "exemplarCity": "ונקובר"
    },
    "Whitehorse": {
      "exemplarCity": "ווייטהורס"
    },
    "Winnipeg": {
      "exemplarCity": "וויניפג"
    },
    "Yakutat": {
      "exemplarCity": "יקוטאט"
    },
    "Yellowknife": {
      "exemplarCity": "ילונייף"
    }
  },
  "Atlantic": {
    "Azores": {
      "exemplarCity": "האיים האזוריים"
    },
    "Bermuda": {
      "exemplarCity": "ברמודה"
    },
    "Canary": {
      "exemplarCity": "האיים הקנריים"
    },
    "Cape_Verde": {
      "exemplarCity": "כף ורדה"
    },
    "Faeroe": {
      "exemplarCity": "פארו"
    },
    "Madeira": {
      "exemplarCity": "מדיירה"
    },
    "Reykjavik": {
      "exemplarCity": "רייקיאוויק"
    },
    "South_Georgia": {
      "exemplarCity": "דרום ג'ורג'יה"
    },
    "St_Helena": {

```



```

        "exemplarCity": "סנט הלנה"
    },
    "Stanley": {
        "exemplarCity": "סטנלי"
    }
},
"Europe": {
    "Amsterdam": {
        "exemplarCity": "אמסטרדם"
    },
    "Andorra": {
        "exemplarCity": "אנדורה"
    },
    "Astrakhan": {
        "exemplarCity": "אסטרן"
    },
    "Athens": {
        "exemplarCity": "אתונה"
    },
    "Belgrade": {
        "exemplarCity": "בלגרד"
    },
    "Berlin": {
        "exemplarCity": "ברלין"
    },
    "Bratislava": {
        "exemplarCity": "ברטיסלבה"
    },
    "Brussels": {
        "exemplarCity": "בריסל"
    },
    "Bucharest": {
        "exemplarCity": "בוקרשט"
    },
    "Budapest": {
        "exemplarCity": "בודפשט"
    },
    "Busingen": {
        "exemplarCity": "ביזינגן"
    },
    "Chisinau": {
        "exemplarCity": "קיישינב"
    },
    "Copenhagen": {
        "exemplarCity": "קופנהגן"
    },
    "Dublin": {
        "long": {
            "daylight": "שעון קיץ אירלנד"
        },
        "exemplarCity": "דבלין"
    },
    "Gibraltar": {
        "exemplarCity": "גיברלטר"
    },
    "Guernsey": {
        "exemplarCity": "גרנזי"
    }
}

```

```
    },  
    "Helsinki": {  
      "exemplarCity": "הלסינקי"  
    },  
    "Isle_of_Man": {  
      "exemplarCity": "האי מאן"  
    },  
    "Istanbul": {  
      "exemplarCity": "איסטנבול"  
    },  
    "Jersey": {  
      "exemplarCity": "ג'רזי"  
    },  
    "Kaliningrad": {  
      "exemplarCity": "קלינינגרד"  
    },  
    "Kiev": {  
      "exemplarCity": "קייב"  
    },  
    "Kirov": {  
      "exemplarCity": "קירוב"  
    },  
    "Lisbon": {  
      "exemplarCity": "ליסבון"  
    },  
    "Ljubljana": {  
      "exemplarCity": "לובליאנה"  
    },  
    "London": {  
      "long": {  
        "daylight": "שעון קיץ בריטניה"  
      },  
      "exemplarCity": "לונדון"  
    },  
    "Luxembourg": {  
      "exemplarCity": "לוקסמבורג"  
    },  
    "Madrid": {  
      "exemplarCity": "מדריד"  
    },  
    "Malta": {  
      "exemplarCity": "מלטה"  
    },  
    "Mariehamn": {  
      "exemplarCity": "מריהאמן"  
    },  
    "Minsk": {  
      "exemplarCity": "מינסק"  
    },  
    "Monaco": {  
      "exemplarCity": "מונקו"  
    },  
    "Moscow": {  
      "exemplarCity": "מוסקבה"  
    },  
    "Oslo": {  
      "exemplarCity": "אוסלו"
```

```
    },
    "Paris": {
      "exemplarCity": "פריז"
    },
    "Podgorica": {
      "exemplarCity": "פודגוריצה"
    },
    "Prague": {
      "exemplarCity": "פראג"
    },
    "Riga": {
      "exemplarCity": "ריגה"
    },
    "Rome": {
      "exemplarCity": "רומא"
    },
    "Samara": {
      "exemplarCity": "סמרה"
    },
    "San_Marino": {
      "exemplarCity": "סן מרינו"
    },
    "Sarajevo": {
      "exemplarCity": "סרייבו"
    },
    "Simferopol": {
      "exemplarCity": "סימפרופול"
    },
    "Skopje": {
      "exemplarCity": "סקופיה"
    },
    "Sofia": {
      "exemplarCity": "סופיה"
    },
    "Stockholm": {
      "exemplarCity": "שטוקהולם"
    },
    "Tallinn": {
      "exemplarCity": "טאלין"
    },
    "Tirane": {
      "exemplarCity": "טירנה"
    },
    "Ulyanovsk": {
      "exemplarCity": "אוליאנובסק"
    },
    "Uzhgorod": {
      "exemplarCity": "אוז'הורוד"
    },
    "Vaduz": {
      "exemplarCity": "ואדוץ"
    },
    "Vatican": {
      "exemplarCity": "הוותיקן"
    },
    "Vienna": {
      "exemplarCity": "וינה"
    }
  }
```

```
    },  
    "Vilnius": {  
      "exemplarCity": "ווילנה"  
    },  
    "Volgograd": {  
      "exemplarCity": "וולגוגרד"  
    },  
    "Warsaw": {  
      "exemplarCity": "ורשה"  
    },  
    "Zagreb": {  
      "exemplarCity": "זאגרב"  
    },  
    "Zaporozhye": {  
      "exemplarCity": "זפורוז'יה"  
    },  
    "Zurich": {  
      "exemplarCity": "ציריך"  
    }  
  },  
  "Africa": {  
    "Abidjan": {  
      "exemplarCity": "אביג'אן"  
    },  
    "Accra": {  
      "exemplarCity": "אקרה"  
    },  
    "Addis_Ababa": {  
      "exemplarCity": "אדיס אבבה"  
    },  
    "Algiers": {  
      "exemplarCity": "אלג'יר"  
    },  
    "Asmera": {  
      "exemplarCity": "אסמרה"  
    },  
    "Bamako": {  
      "exemplarCity": "במאקו"  
    },  
    "Bangui": {  
      "exemplarCity": "בנגואי"  
    },  
    "Banjul": {  
      "exemplarCity": "בנג'ול"  
    },  
    "Bissau": {  
      "exemplarCity": "ביסאו"  
    },  
    "Blantyre": {  
      "exemplarCity": "בלנטיר"  
    },  
    "Brazzaville": {  
      "exemplarCity": "ברזוויל"  
    },  
    "Bujumbura": {  
      "exemplarCity": "בוג'ומבורה"  
    }  
  },  
}
```

```
"Cairo": {
  "exemplarCity": "קהיר"
},
"Casablanca": {
  "exemplarCity": "קזבלנקה"
},
"Ceuta": {
  "exemplarCity": "סאוטה"
},
"Conakry": {
  "exemplarCity": "קונאקרי"
},
"Dakar": {
  "exemplarCity": "דקאר"
},
"Dar_es_Salaam": {
  "exemplarCity": "דאר א-סלאם"
},
"Djibouti": {
  "exemplarCity": "ג'יבוטי"
},
"Douala": {
  "exemplarCity": "דואלה"
},
"El_Aaiun": {
  "exemplarCity": "אל עיון"
},
"Freetown": {
  "exemplarCity": "פריטאון"
},
"Gaborone": {
  "exemplarCity": "גבורונה"
},
"Harare": {
  "exemplarCity": "הרארה"
},
"Johannesburg": {
  "exemplarCity": "יוהנסבורג"
},
"Juba": {
  "exemplarCity": "ג'ובה"
},
"Kampala": {
  "exemplarCity": "קמפלה"
},
"Khartoum": {
  "exemplarCity": "חרטום"
},
"Kigali": {
  "exemplarCity": "קיגלי"
},
"Kinshasa": {
  "exemplarCity": "קינשה"
},
"Lagos": {
  "exemplarCity": "לגוס"
},
}
```

```
"Libreville": {
  "exemplarCity": "ליברוויל"
},
"Lome": {
  "exemplarCity": "לומה"
},
"Luanda": {
  "exemplarCity": "לואנדה"
},
"Lubumbashi": {
  "exemplarCity": "לובומבאשי"
},
"Lusaka": {
  "exemplarCity": "לוסקה"
},
"Malabo": {
  "exemplarCity": "מלבו"
},
"Maputo": {
  "exemplarCity": "מאפוטו"
},
"Maseru": {
  "exemplarCity": "מסרו"
},
"Mbabane": {
  "exemplarCity": "אמבאבאנה"
},
"Mogadishu": {
  "exemplarCity": "מוגדישו"
},
"Monrovia": {
  "exemplarCity": "מונרוביה"
},
"Nairobi": {
  "exemplarCity": "ניירובי"
},
"Ndjamena": {
  "exemplarCity": "נג'מנה"
},
"Niamey": {
  "exemplarCity": "ניאמי"
},
"Nouakchott": {
  "exemplarCity": "נואקצ'וט"
},
"Ouagadougou": {
  "exemplarCity": "וואגאדוגו"
},
"Porto-Novo": {
  "exemplarCity": "פורטו נובו"
},
"Sao_Tome": {
  "exemplarCity": "סאו טומה"
},
"Tripoli": {
  "exemplarCity": "טריפולי"
},
}
```

```
"Tunis": {
  "exemplarCity": "תוניס"
},
"Windhoek": {
  "exemplarCity": "ווינדהוק"
}
},
"Asia": {
  "Aden": {
    "exemplarCity": "עדן"
  },
  "Almaty": {
    "exemplarCity": "אלמאטי"
  },
  "Amman": {
    "exemplarCity": "עמאן"
  },
  "Anadyr": {
    "exemplarCity": "אנדיר"
  },
  "Aqtan": {
    "exemplarCity": "אקטאן"
  },
  "Aqtobe": {
    "exemplarCity": "אקטובה"
  },
  "Ashgabat": {
    "exemplarCity": "אשגבט"
  },
  "Baghdad": {
    "exemplarCity": "בגדד"
  },
  "Bahrain": {
    "exemplarCity": "בחריין"
  },
  "Baku": {
    "exemplarCity": "באקו"
  },
  "Bangkok": {
    "exemplarCity": "בנגקוק"
  },
  "Barnaul": {
    "exemplarCity": "ברנאול"
  },
  "Beirut": {
    "exemplarCity": "ביירות"
  },
  "Bishkek": {
    "exemplarCity": "בישקק"
  },
  "Brunei": {
    "exemplarCity": "ברוניי"
  },
  "Calcutta": {
    "exemplarCity": "קולקטה"
  },
  "Chita": {
```

```
    "exemplarCity": "צ'יטה"
  },
  "Choibalsan": {
    "exemplarCity": "צ'ויבלסן"
  },
  "Colombo": {
    "exemplarCity": "קולומבו"
  },
  "Damascus": {
    "exemplarCity": "דמשק"
  },
  "Dhaka": {
    "exemplarCity": "דאקה"
  },
  "Dili": {
    "exemplarCity": "דילי"
  },
  "Dubai": {
    "exemplarCity": "דובאי"
  },
  "Dushanbe": {
    "exemplarCity": "דושנבה"
  },
  "Gaza": {
    "exemplarCity": "עזה"
  },
  "Hebron": {
    "exemplarCity": "חברון"
  },
  "Hong_Kong": {
    "exemplarCity": "הונג קונג"
  },
  "Hovd": {
    "exemplarCity": "חובד"
  },
  "Irkutsk": {
    "exemplarCity": "אירקוטסק"
  },
  "Jakarta": {
    "exemplarCity": "ג'קרטה"
  },
  "Jayapura": {
    "exemplarCity": "ג'יאפורה"
  },
  "Jerusalem": {
    "exemplarCity": "ירושלים"
  },
  "Kabul": {
    "exemplarCity": "קאבול"
  },
  "Kamchatka": {
    "exemplarCity": "קמצ'טקה"
  },
  "Karachi": {
    "exemplarCity": "קראצ'י"
  },
  "Katmandu": {
```



```

    "exemplarCity": "קטמנדו"
  },
  "Khandyga": {
    "exemplarCity": "חנדיגה"
  },
  "Krasnoyarsk": {
    "exemplarCity": "קרסנויארסק"
  },
  "Kuala_Lumpur": {
    "exemplarCity": "קואלה לומפור"
  },
  "Kuching": {
    "exemplarCity": "קוצ'ינג"
  },
  "Kuwait": {
    "exemplarCity": "כווית"
  },
  "Macau": {
    "exemplarCity": "מקאו"
  },
  "Magadan": {
    "exemplarCity": "מגדן"
  },
  "Makassar": {
    "exemplarCity": "מאקאסאר"
  },
  "Manila": {
    "exemplarCity": "מנילה"
  },
  "Muscat": {
    "exemplarCity": "מוסקט"
  },
  "Nicosia": {
    "exemplarCity": "ניקוסיה"
  },
  "Novokuznetsk": {
    "exemplarCity": "נובוקוזנטסק"
  },
  "Novosibirsk": {
    "exemplarCity": "נובוסירסק"
  },
  "Omsk": {
    "exemplarCity": "אומסק"
  },
  "Oral": {
    "exemplarCity": "אורל"
  },
  "Phnom_Penh": {
    "exemplarCity": "פנום פן"
  },
  "Pontianak": {
    "exemplarCity": "פונטיאנק"
  },
  "Pyongyang": {
    "exemplarCity": "פיונגיאנג"
  },
  "Qatar": {

```

```
    "exemplarCity": "קטאר"  
  },  
  "Qyzylorda": {  
    "exemplarCity": "קזיזילורדה"  
  },  
  "Rangoon": {  
    "exemplarCity": "רנגון"  
  },  
  "Riyadh": {  
    "exemplarCity": "ריאד"  
  },  
  "Saigon": {  
    "exemplarCity": "הו צ'י מין סיטי"  
  },  
  "Sakhalin": {  
    "exemplarCity": "סחלין"  
  },  
  "Samarkand": {  
    "exemplarCity": "סמרקנד"  
  },  
  "Seoul": {  
    "exemplarCity": "סיאול"  
  },  
  "Shanghai": {  
    "exemplarCity": "שנחאי"  
  },  
  "Singapore": {  
    "exemplarCity": "סינגפור"  
  },  
  "Srednekolymsk": {  
    "exemplarCity": "סרדנייקולימסק"  
  },  
  "Taipei": {  
    "exemplarCity": "טאיפיי"  
  },  
  "Tashkent": {  
    "exemplarCity": "טשקנט"  
  },  
  "Tbilisi": {  
    "exemplarCity": "טביליסי"  
  },  
  "Tehran": {  
    "exemplarCity": "טהרן"  
  },  
  "Thimphu": {  
    "exemplarCity": "טהימפהו"  
  },  
  "Tokyo": {  
    "exemplarCity": "טוקיו"  
  },  
  "Tomsk": {  
    "exemplarCity": "טומסק"  
  },  
  "Ulaanbaatar": {  
    "exemplarCity": "אולאאנבטאר"  
  },  
  "Urumqi": {
```

```

        "exemplarCity": "אורומקי"
      },
      "Ust-Nera": {
        "exemplarCity": "אוסט-נרה"
      },
      "Vientiane": {
        "exemplarCity": "האנוי"
      },
      "Vladivostok": {
        "exemplarCity": "ולדיווסטוק"
      },
      "Yakutsk": {
        "exemplarCity": "יקוטסק"
      },
      "Yekaterinburg": {
        "exemplarCity": "יקטרינבורג"
      },
      "Yerevan": {
        "exemplarCity": "ירוואן"
      }
    },
    "Indian": {
      "Antananarivo": {
        "exemplarCity": "אנטננריבו"
      },
      "Chagos": {
        "exemplarCity": "צ'אגוס"
      },
      "Christmas": {
        "exemplarCity": "האי כריסטמס"
      },
      "Cocos": {
        "exemplarCity": "קוקוס"
      },
      "Comoro": {
        "exemplarCity": "קומורו"
      },
      "Kerguelen": {
        "exemplarCity": "קרגוולן"
      },
      "Mahe": {
        "exemplarCity": "מהא"
      },
      "Maldives": {
        "exemplarCity": "האיים המלדיביים"
      },
      "Mauritius": {
        "exemplarCity": "מאוריציוס"
      },
      "Mayotte": {
        "exemplarCity": "מאיוט"
      },
      "Reunion": {
        "exemplarCity": "ראוניון"
      }
    },
    "Australia": {

```

```
"Adelaide": {
  "exemplarCity": "אדלייד"
},
"Brisbane": {
  "exemplarCity": "בריסביין"
},
"Broken_Hill": {
  "exemplarCity": "ברוקן היל"
},
"Currie": {
  "exemplarCity": "קרי"
},
"Darwin": {
  "exemplarCity": "דרווין"
},
"Eucla": {
  "exemplarCity": "יוקלה"
},
"Hobart": {
  "exemplarCity": "הוברט"
},
"Lindeman": {
  "exemplarCity": "לינדמן"
},
"Lord_Howe": {
  "exemplarCity": "אי הלורד האו"
},
"Melbourne": {
  "exemplarCity": "מלבורן"
},
"Perth": {
  "exemplarCity": "פרת' "
},
"Sydney": {
  "exemplarCity": "סידני"
}
},
"Pacific": {
  "Apia": {
    "exemplarCity": "אפיה"
  },
  "Auckland": {
    "exemplarCity": "אוקלנד"
  },
  "Bougainville": {
    "exemplarCity": "בוגנוויל"
  },
  "Chatham": {
    "exemplarCity": "צ'אטהאם"
  },
  "Easter": {
    "exemplarCity": "אי הפסחא"
  },
  "Efate": {
    "exemplarCity": "אפטה"
  },
  "Enderbury": {
```

```
    "exemplarCity": "אנדוררי"  
  },  
  "Fakaofo": {  
    "exemplarCity": "פקאופו"  
  },  
  "Fiji": {  
    "exemplarCity": "פיג'י"  
  },  
  "Funafuti": {  
    "exemplarCity": "פונפוט"י"  
  },  
  "Galapagos": {  
    "exemplarCity": "גלפאגוס"  
  },  
  "Gambier": {  
    "exemplarCity": "איי גמבייה"  
  },  
  "Guadalcanal": {  
    "exemplarCity": "גוודלקנאל"  
  },  
  "Guam": {  
    "exemplarCity": "גואם"  
  },  
  "Honolulu": {  
    "exemplarCity": "הונוולו"  
  },  
  "Johnston": {  
    "exemplarCity": "ג'ונסטון"  
  },  
  "Kiritimati": {  
    "exemplarCity": "קיריטימאטי"  
  },  
  "Kosrae": {  
    "exemplarCity": "קוסרה"  
  },  
  "Kwajalein": {  
    "exemplarCity": "קוואג'ליין"  
  },  
  "Majuro": {  
    "exemplarCity": "מאג'ורו"  
  },  
  "Marquesas": {  
    "exemplarCity": "איי מרקז"  
  },  
  "Midway": {  
    "exemplarCity": "מידוויי"  
  },  
  "Nauru": {  
    "exemplarCity": "נאורו"  
  },  
  "Niue": {  
    "exemplarCity": "ניואה"  
  },  
  "Norfolk": {  
    "exemplarCity": "נורפוק"  
  },  
  "Noumea": {
```

```

        "exemplarCity": "נומאה"
    },
    "Pago_Pago": {
        "exemplarCity": "פאגו פאגו"
    },
    "Palau": {
        "exemplarCity": "פלאו"
    },
    "Pitcairn": {
        "exemplarCity": "פיטקרן"
    },
    "Ponape": {
        "exemplarCity": "פונפיי"
    },
    "Port_Moresby": {
        "exemplarCity": "פורט מורסבי"
    },
    "Rarotonga": {
        "exemplarCity": "רארוטונגה"
    },
    "Saipan": {
        "exemplarCity": "סאיפאן"
    },
    "Tahiti": {
        "exemplarCity": "טהיטי"
    },
    "Tarawa": {
        "exemplarCity": "טאראווה"
    },
    "Tongatapu": {
        "exemplarCity": "טונגטאפו"
    },
    "Truk": {
        "exemplarCity": "צ'וק"
    },
    "Wake": {
        "exemplarCity": "ווייק"
    },
    "Wallis": {
        "exemplarCity": "ווליס"
    }
},
"Arctic": {
    "Longyearbyen": {
        "exemplarCity": "לונגיירבין"
    }
},
"Antarctica": {
    "Casey": {
        "exemplarCity": "קאסיי"
    },
    "Davis": {
        "exemplarCity": "דייוויס"
    },
    "DumontDUrville": {
        "exemplarCity": "דומון ד'אורוויל"
    }
},

```

```
"Macquarie": {
  "exemplarCity": "מקרי"
},
"Mawson": {
  "exemplarCity": "מוסון"
},
"McMurdo": {
  "exemplarCity": "מק-מרדו"
},
"Palmer": {
  "exemplarCity": "פאלמר"
},
"Rothera": {
  "exemplarCity": "רות'רה"
},
"Syowa": {
  "exemplarCity": "סיוואה"
},
"Troll": {
  "exemplarCity": "טרול"
},
"Vostok": {
  "exemplarCity": "ווסטוק"
}
},
"Etc": {
  "GMT": {
    "exemplarCity": "GMT"
  },
  "GMT1": {
    "exemplarCity": "GMT+1"
  },
  "GMT10": {
    "exemplarCity": "GMT+10"
  },
  "GMT11": {
    "exemplarCity": "GMT+11"
  },
  "GMT12": {
    "exemplarCity": "GMT+12"
  },
  "GMT2": {
    "exemplarCity": "GMT+2"
  },
  "GMT3": {
    "exemplarCity": "GMT+3"
  },
  "GMT4": {
    "exemplarCity": "GMT+4"
  },
  "GMT5": {
    "exemplarCity": "GMT+5"
  },
  "GMT6": {
    "exemplarCity": "GMT+6"
  },
  "GMT7": {
```

```

    "exemplarCity": "GMT+7"
  },
  "GMT8": {
    "exemplarCity": "GMT+8"
  },
  "GMT9": {
    "exemplarCity": "GMT+9"
  },
  "GMT-1": {
    "exemplarCity": "GMT-1"
  },
  "GMT-10": {
    "exemplarCity": "GMT-10"
  },
  "GMT-11": {
    "exemplarCity": "GMT-11"
  },
  "GMT-12": {
    "exemplarCity": "GMT-12"
  },
  "GMT-13": {
    "exemplarCity": "GMT-13"
  },
  "GMT-14": {
    "exemplarCity": "GMT-14"
  },
  "GMT-2": {
    "exemplarCity": "GMT-2"
  },
  "GMT-3": {
    "exemplarCity": "GMT-3"
  },
  "GMT-4": {
    "exemplarCity": "GMT-4"
  },
  "GMT-5": {
    "exemplarCity": "GMT-5"
  },
  "GMT-6": {
    "exemplarCity": "GMT-6"
  },
  "GMT-7": {
    "exemplarCity": "GMT-7"
  },
  "GMT-8": {
    "exemplarCity": "GMT-8"
  },
  "GMT-9": {
    "exemplarCity": "GMT-9"
  },
  "UTC": {
    "long": {
      "standard": "זמן אוניברסלי מתואם"
    },
    "short": {
      "standard": "UTC"
    }
  },

```



```

        "exemplarCity": "UTC"
      },
      "Unknown": {
        "exemplarCity": "עיר לא ידועה"
      }
    },
    "metazone": {
      "Afghanistan": {
        "long": {
          "standard": "שעון אפגניסטן"
        }
      },
      "Africa_Central": {
        "long": {
          "standard": "שעון מרכז אפריקה"
        }
      },
      "Africa_Eastern": {
        "long": {
          "standard": "שעון מזרח אפריקה"
        }
      },
      "Africa_Southern": {
        "long": {
          "standard": "שעון דרום אפריקה"
        }
      },
      "Africa_Western": {
        "long": {
          "generic": "שעון מערב אפריקה",
          "standard": "שעון מערב אפריקה (חורף)",
          "daylight": "שעון מערב אפריקה (קיץ)"
        }
      },
      "Alaska": {
        "long": {
          "generic": "שעון אלסקה",
          "standard": "שעון אלסקה (חורף)",
          "daylight": "שעון אלסקה (קיץ)"
        }
      },
      "Amazon": {
        "long": {
          "generic": "שעון אמזונס",
          "standard": "שעון אמזונס (חורף)",
          "daylight": "שעון אמזונס (קיץ)"
        }
      },
      "America_Central": {
        "long": {
          "generic": "שעון מרכז ארה״ב",
          "standard": "שעון מרכז ארה״ב (חורף)",
          "daylight": "שעון מרכז ארה״ב (קיץ)"
        }
      },
      "America_Eastern": {

```

```

    "long": {
      "generic": "שעון החוף המזרחי",
      "standard": "שעון החוף המזרחי (חורף)",
      "daylight": "שעון החוף המזרחי (קיץ)"
    }
  },
  "America_Mountain": {
    "long": {
      "generic": "שעון אזור ההרים בארה"ב",
      "standard": "שעון אזור ההרים בארה"ב (חורף)",
      "daylight": "שעון אזור ההרים בארה"ב (קיץ)"
    }
  },
  "America_Pacific": {
    "long": {
      "generic": "שעון מערב ארה"ב",
      "standard": "שעון מערב ארה"ב (חורף)",
      "daylight": "שעון מערב ארה"ב (קיץ)"
    }
  },
  "Anadyr": {
    "long": {
      "generic": "שעון אנדיר",
      "standard": "שעון רגיל אנדיר",
      "daylight": "שעון קיץ אנדיר"
    }
  },
  "Apia": {
    "long": {
      "generic": "שעון אפיה",
      "standard": "שעון אפיה (חורף)",
      "daylight": "שעון אפיה (קיץ)"
    }
  },
  "Arabian": {
    "long": {
      "generic": "שעון חצי האי ערב",
      "standard": "שעון חצי האי ערב (חורף)",
      "daylight": "שעון חצי האי ערב (קיץ)"
    }
  },
  "Argentina": {
    "long": {
      "generic": "שעון ארגנטינה",
      "standard": "שעון ארגנטינה (חורף)",
      "daylight": "שעון ארגנטינה (קיץ)"
    }
  },
  "Argentina_Western": {
    "long": {
      "generic": "שעון מערב ארגנטינה",
      "standard": "שעון מערב ארגנטינה (חורף)",
      "daylight": "שעון מערב ארגנטינה (קיץ)"
    }
  },
  "Armenia": {
    "long": {

```

```

        "generic": "שעון ארמניה",
        "standard": "שעון ארמניה (חורף)",
        "daylight": "שעון ארמניה (קיץ)"
    },
},
"Atlantic": {
    "long": {
        "generic": "שעון האוקיינוס האטלנטי",
        "standard": "שעון האוקיינוס האטלנטי (חורף)",
        "daylight": "שעון האוקיינוס האטלנטי (קיץ)"
    }
},
"Australia_Central": {
    "long": {
        "generic": "שעון מרכז אוסטרליה",
        "standard": "שעון מרכז אוסטרליה (חורף)",
        "daylight": "שעון מרכז אוסטרליה (קיץ)"
    }
},
"Australia_CentralWestern": {
    "long": {
        "generic": "שעון מרכז-מערב אוסטרליה",
        "standard": "שעון מרכז-מערב אוסטרליה (חורף)",
        "daylight": "שעון מרכז-מערב אוסטרליה (קיץ)"
    }
},
"Australia_Eastern": {
    "long": {
        "generic": "שעון מזרח אוסטרליה",
        "standard": "שעון מזרח אוסטרליה (חורף)",
        "daylight": "שעון מזרח אוסטרליה (קיץ)"
    }
},
"Australia_Western": {
    "long": {
        "generic": "שעון מערב אוסטרליה",
        "standard": "שעון מערב אוסטרליה (חורף)",
        "daylight": "שעון מערב אוסטרליה (קיץ)"
    }
},
"Azerbaijan": {
    "long": {
        "generic": "שעון אזרבייג'אן",
        "standard": "שעון אזרבייג'אן (חורף)",
        "daylight": "שעון אזרבייג'אן (קיץ)"
    }
},
"Azores": {
    "long": {
        "generic": "שעון האיים האזוריים",
        "standard": "שעון האיים האזוריים (חורף)",
        "daylight": "שעון האיים האזוריים (קיץ)"
    }
},
"Bangladesh": {
    "long": {
        "generic": "שעון בנגלדש",

```

```

        "standard": "שעון בנגלדש (חורף)",
        "daylight": "שעון בנגלדש (קיץ)"
    },
    },
    "Bhutan": {
        "long": {
            "standard": "שעון בהוטן"
        }
    },
    },
    "Bolivia": {
        "long": {
            "standard": "שעון בוליביה"
        }
    },
    },
    "Brasilia": {
        "long": {
            "generic": "שעון ברזיליה",
            "standard": "שעון ברזיליה (חורף)",
            "daylight": "שעון ברזיליה (קיץ)"
        }
    },
    },
    "Brunei": {
        "long": {
            "standard": "שעון ברוניי דארוסלאם"
        }
    },
    },
    "Cape_Verde": {
        "long": {
            "generic": "שעון כף ורדה",
            "standard": "שעון כף ורדה (חורף)",
            "daylight": "שעון כף ורדה (קיץ)"
        }
    },
    },
    "Chamorro": {
        "long": {
            "standard": "שעון צ'אמורו"
        }
    },
    },
    "Chatham": {
        "long": {
            "generic": "שעון צ'טהאם",
            "standard": "שעון צ'טהאם (חורף)",
            "daylight": "שעון צ'טהאם (קיץ)"
        }
    },
    },
    "Chile": {
        "long": {
            "generic": "שעון צ'ילה",
            "standard": "שעון צ'ילה (חורף)",
            "daylight": "שעון צ'ילה (קיץ)"
        }
    },
    },
    "China": {
        "long": {
            "generic": "שעון סין",
            "standard": "שעון סין (חורף)",
            "daylight": "שעון סין (קיץ)"
        }
    }

```

```

    }
  },
  "Choibalsan": {
    "long": {
      "generic": "שעון צ'ויבלסן",
      "standard": "שעון צ'ויבלסן (חורף)",
      "daylight": "שעון צ'ויבלסן (קיץ)"
    }
  },
  "Christmas": {
    "long": {
      "standard": "שעון האי כריסטמס"
    }
  },
  "Cocos": {
    "long": {
      "standard": "שעון איי קוקוס"
    }
  },
  "Colombia": {
    "long": {
      "generic": "שעון קולומביה",
      "standard": "שעון קולומביה (חורף)",
      "daylight": "שעון קולומביה (קיץ)"
    }
  },
  "Cook": {
    "long": {
      "generic": "שעון איי קוק",
      "standard": "שעון איי קוק (חורף)",
      "daylight": "שעון איי קוק (מחצית הקיץ)"
    }
  },
  "Cuba": {
    "long": {
      "generic": "שעון קובה",
      "standard": "שעון קובה (חורף)",
      "daylight": "שעון קובה (קיץ)"
    }
  },
  "Davis": {
    "long": {
      "standard": "שעון דייוויס"
    }
  },
  "DumontDUrville": {
    "long": {
      "standard": "שעון דומון ד'אורוויל"
    }
  },
  "East_Timor": {
    "long": {
      "standard": "שעון מזרח טימור"
    }
  },
  "Easter": {
    "long": {

```

```

        "generic": "שעון אי הפסחא",
        "standard": "שעון אי הפסחא (חורף)",
        "daylight": "שעון אי הפסחא (קיץ)"
    }
},
"Ecuador": {
    "long": {
        "standard": "שעון אקוודור"
    }
},
"Europe_Central": {
    "long": {
        "generic": "שעון מרכז אירופה",
        "standard": "שעון מרכז אירופה (חורף)",
        "daylight": "שעון מרכז אירופה (קיץ)"
    }
},
"Europe_Eastern": {
    "long": {
        "generic": "שעון מזרח אירופה",
        "standard": "שעון מזרח אירופה (חורף)",
        "daylight": "שעון מזרח אירופה (קיץ)"
    }
},
"Europe_Further_Eastern": {
    "long": {
        "standard": "שעון מינסק"
    }
},
"Europe_Western": {
    "long": {
        "generic": "שעון מערב אירופה",
        "standard": "שעון מערב אירופה (חורף)",
        "daylight": "שעון מערב אירופה (קיץ)"
    }
},
"Falkland": {
    "long": {
        "generic": "שעון איי פוקלנד",
        "standard": "שעון איי פוקלנד (חורף)",
        "daylight": "שעון איי פוקלנד (קיץ)"
    }
},
"Fiji": {
    "long": {
        "generic": "שעון פיג'י",
        "standard": "שעון פיג'י (חורף)",
        "daylight": "שעון פיג'י (קיץ)"
    }
},
"French_Guiana": {
    "long": {
        "standard": "שעון גיאנה הצרפתית"
    }
},
"French_Southern": {
    "long": {

```

```

        "standard": "שעון הארצות הדרומיות והאנטארקטיות של צרפת"
    },
    },
    "Galapagos": {
        "long": {
            "standard": "שעון איי גלאפגוס"
        }
    },
    },
    "Gambier": {
        "long": {
            "standard": "שעון איי גמבייה"
        }
    },
    },
    "Georgia": {
        "long": {
            "generic": "שעון גאורגיה",
            "standard": "שעון גאורגיה (חורף)",
            "daylight": "שעון גאורגיה (קיץ)"
        }
    },
    },
    "Gilbert_Islands": {
        "long": {
            "standard": "שעון איי גילברט"
        }
    },
    },
    "GMT": {
        "long": {
            "standard": "שעון גריניץ'"
        }
    },
    },
    "Greenland_Eastern": {
        "long": {
            "generic": "שעון מזרח גרינלנד",
            "standard": "שעון מזרח גרינלנד (חורף)",
            "daylight": "שעון מזרח גרינלנד (קיץ)"
        }
    },
    },
    "Greenland_Western": {
        "long": {
            "generic": "שעון מערב גרינלנד",
            "standard": "שעון מערב גרינלנד (חורף)",
            "daylight": "שעון מערב גרינלנד (קיץ)"
        }
    },
    },
    "Gulf": {
        "long": {
            "standard": "שעון מדינות המפרץ"
        }
    },
    },
    "Guyana": {
        "long": {
            "standard": "שעון גיאנה"
        }
    },
    },
    "Hawaii_Aleutian": {
        "long": {
            "generic": "שעון האיים האלאוטיים הוואיי",

```

```

        "standard": "שעון האיים האלאוטיים הוואי (חורף)",
        "daylight": "שעון האיים האלאוטיים הוואי (קיץ)"
    },
    "Hong_Kong": {
        "long": {
            "generic": "שעון הונג קונג",
            "standard": "שעון הונג קונג (חורף)",
            "daylight": "שעון הונג קונג (קיץ)"
        }
    },
    "Hovd": {
        "long": {
            "generic": "שעון חובד",
            "standard": "שעון חובד (חורף)",
            "daylight": "שעון חובד (קיץ)"
        }
    },
    "India": {
        "long": {
            "standard": "שעון הודו"
        }
    },
    "Indian_Ocean": {
        "long": {
            "standard": "שעון האוקיינוס ההודי"
        }
    },
    "Indochina": {
        "long": {
            "standard": "שעון הודו-סין"
        }
    },
    "Indonesia_Central": {
        "long": {
            "standard": "שעון מרכז אינדונזיה"
        }
    },
    "Indonesia_Eastern": {
        "long": {
            "standard": "שעון מזרח אינדונזיה"
        }
    },
    "Indonesia_Western": {
        "long": {
            "standard": "שעון מערב אינדונזיה"
        }
    },
    "Iran": {
        "long": {
            "generic": "שעון איראן",
            "standard": "שעון איראן (חורף)",
            "daylight": "שעון איראן (קיץ)"
        }
    },
    "Irkutsk": {
        "long": {

```



```

        "generic": "שעון אירקוטסק",
        "standard": "שעון אירקוטסק (חורף)",
        "daylight": "שעון אירקוטסק (קיץ)"
    },
},
"Israel": {
    "long": {
        "generic": "שעון ישראל",
        "standard": "שעון ישראל (חורף)",
        "daylight": "שעון ישראל (קיץ)"
    }
},
"Japan": {
    "long": {
        "generic": "שעון יפן",
        "standard": "שעון יפן (חורף)",
        "daylight": "שעון יפן (קיץ)"
    }
},
"Kamchatka": {
    "long": {
        "generic": "שעון פטרופלובסק-קמצ'טסקי",
        "standard": "שעון רגיל פטרופלובסק-קמצ'טסקי",
        "daylight": "שעון קיץ פטרופלובסק-קמצ'טסקי"
    }
},
"Kazakhstan_Eastern": {
    "long": {
        "standard": "שעון מזרח קזחסטן"
    }
},
"Kazakhstan_Western": {
    "long": {
        "standard": "שעון מערב קזחסטן"
    }
},
"Korea": {
    "long": {
        "generic": "שעון קוריאה",
        "standard": "שעון קוריאה (חורף)",
        "daylight": "שעון קוריאה (קיץ)"
    }
},
"Kosrae": {
    "long": {
        "standard": "שעון קוסראה"
    }
},
"Krasnoyarsk": {
    "long": {
        "generic": "שעון קרסנויארסק",
        "standard": "שעון קרסנויארסק (חורף)",
        "daylight": "שעון קרסנויארסק (קיץ)"
    }
},
"Kyrgystan": {
    "long": {

```

```

        "standard": "שעון קירגיזסטן"
    },
    },
    "Line_Islands": {
        "long": {
            "standard": "שעון איי ליין"
        }
    },
    },
    "Lord_Howe": {
        "long": {
            "generic": "שעון אי הלורד האו",
            "standard": "שעון אי הלורד האו (חורף)",
            "daylight": "שעון אי הלורד האו (קיץ)"
        }
    },
    },
    "Macau": {
        "long": {
            "generic": "שעון מקאו",
            "standard": "שעון חורף מקאו",
            "daylight": "שעון קיץ מקאו"
        }
    },
    },
    "Macquarie": {
        "long": {
            "standard": "שעון מקווארי"
        }
    },
    },
    "Magadan": {
        "long": {
            "generic": "שעון מגדן",
            "standard": "שעון מגדן (חורף)",
            "daylight": "שעון מגדן (קיץ)"
        }
    },
    },
    "Malaysia": {
        "long": {
            "standard": "שעון מלזיה"
        }
    },
    },
    "Maldives": {
        "long": {
            "standard": "שעון האיים המלדיביים"
        }
    },
    },
    "Marquesas": {
        "long": {
            "standard": "שעון איי מרקז"
        }
    },
    },
    "Marshall_Islands": {
        "long": {
            "standard": "שעון איי מרשל"
        }
    },
    },
    "Mauritius": {
        "long": {
            "generic": "שעון מאוריציוס",

```

```

        "standard": "שעון מאוריציז (חורף)",
        "daylight": "שעון מאוריציז (קיץ)"
    },
},
"Mawson": {
    "long": {
        "standard": "שעון מאוסון"
    }
},
"Mexico_Northwest": {
    "long": {
        "generic": "שעון צפון-מערב מקסיקו",
        "standard": "שעון צפון-מערב מקסיקו (חורף)",
        "daylight": "שעון צפון-מערב מקסיקו (קיץ)"
    }
},
"Mexico_Pacific": {
    "long": {
        "generic": "שעון מערב מקסיקו",
        "standard": "שעון מערב מקסיקו (חורף)",
        "daylight": "שעון מערב מקסיקו (קיץ)"
    }
},
"Mongolia": {
    "long": {
        "generic": "שעון אולן בטור",
        "standard": "שעון אולן בטור (חורף)",
        "daylight": "שעון אולן בטור (קיץ)"
    }
},
"Moscow": {
    "long": {
        "generic": "שעון מוסקבה",
        "standard": "שעון מוסקבה (חורף)",
        "daylight": "שעון מוסקבה (קיץ)"
    }
},
"Myanmar": {
    "long": {
        "standard": "שעון מיאנמר"
    }
},
"Nauru": {
    "long": {
        "standard": "שעון נאורו"
    }
},
"Nepal": {
    "long": {
        "standard": "שעון נפאל"
    }
},
"New_Caledonia": {
    "long": {
        "generic": "שעון קלדוניה החדשה",
        "standard": "שעון קלדוניה החדשה (חורף)",
        "daylight": "שעון קלדוניה החדשה (קיץ)"
    }
}

```

```
    },
    "New_Zealand": {
      "long": {
        "generic": "שעון ניו זילנד",
        "standard": "שעון ניו זילנד (חורף)",
        "daylight": "שעון ניו זילנד (קיץ)"
      }
    },
    "Newfoundland": {
      "long": {
        "generic": "שעון ניופאונדלנד",
        "standard": "שעון ניופאונדלנד (חורף)",
        "daylight": "שעון ניופאונדלנד (קיץ)"
      }
    },
    "Niue": {
      "long": {
        "standard": "שעון ניואה"
      }
    },
    "Norfolk": {
      "long": {
        "standard": "שעון האי נורפוק"
      }
    },
    "Noronha": {
      "long": {
        "generic": "שעון פרננדו די נורוניה",
        "standard": "שעון פרננדו די נורוניה (חורף)",
        "daylight": "שעון פרננדו די נורוניה (קיץ)"
      }
    },
    "Novosibirsk": {
      "long": {
        "generic": "שעון נובוסיבירסק",
        "standard": "שעון נובוסיבירסק (חורף)",
        "daylight": "שעון נובוסיבירסק (קיץ)"
      }
    },
    "Omsk": {
      "long": {
        "generic": "שעון אומסק",
        "standard": "שעון אומסק (חורף)",
        "daylight": "שעון אומסק (קיץ)"
      }
    },
    "Pakistan": {
      "long": {
        "generic": "שעון פקיסטן",
        "standard": "שעון פקיסטן (חורף)",
        "daylight": "שעון פקיסטן (קיץ)"
      }
    },
    "Palau": {
      "long": {
        "standard": "שעון פלאו"
```

```

    }
  },
  "Papua_New_Guinea": {
    "long": {
      "standard": "שעון פפואה גיניאה החדשה"
    }
  },
  "Paraguay": {
    "long": {
      "generic": "שעון פרגוואי",
      "standard": "שעון פרגוואי (חורף)",
      "daylight": "שעון פרגוואי (קיץ)"
    }
  },
  "Peru": {
    "long": {
      "generic": "שעון פרו",
      "standard": "שעון פרו (חורף)",
      "daylight": "שעון פרו (קיץ)"
    }
  },
  "Philippines": {
    "long": {
      "generic": "שעון הפיליפינים",
      "standard": "שעון הפיליפינים (חורף)",
      "daylight": "שעון הפיליפינים (קיץ)"
    }
  },
  "Phoenix_Islands": {
    "long": {
      "standard": "שעון איי פיניקס"
    }
  },
  "Pierre_Miquelon": {
    "long": {
      "generic": "שעון סנט פייר ומיקלון",
      "standard": "שעון סנט פייר ומיקלון (חורף)",
      "daylight": "שעון סנט פייר ומיקלון (קיץ)"
    }
  },
  "Pitcairn": {
    "long": {
      "standard": "שעון פיטקרן"
    }
  },
  "Ponape": {
    "long": {
      "standard": "שעון פונאפי"
    }
  },
  "Pyongyang": {
    "long": {
      "standard": "שעון פיונגיאנג"
    }
  },
  "Reunion": {
    "long": {

```

```

        "standard": "שעון ראוניון"
    },
    },
    "Rothera": {
        "long": {
            "standard": "שעון רות'רה"
        }
    },
    "Sakhalin": {
        "long": {
            "generic": "שעון סחלין",
            "standard": "שעון סחלין (חורף)",
            "daylight": "שעון סחלין (קיץ)"
        }
    },
    "Samara": {
        "long": {
            "generic": "שעון סמרה",
            "standard": "שעון רגיל סמרה",
            "daylight": "שעון קיץ סמרה"
        }
    },
    "Samoa": {
        "long": {
            "generic": "שעון סמואה",
            "standard": "שעון סמואה (חורף)",
            "daylight": "שעון סמואה (קיץ)"
        }
    },
    "Seychelles": {
        "long": {
            "standard": "שעון איי סיישל"
        }
    },
    "Singapore": {
        "long": {
            "standard": "שעון סינגפור"
        }
    },
    "Solomon": {
        "long": {
            "standard": "שעון איי שלמה"
        }
    },
    "South_Georgia": {
        "long": {
            "standard": "שעון דרום ג'ורג'יה"
        }
    },
    "Suriname": {
        "long": {
            "standard": "שעון סורינאם"
        }
    },
    "Syowa": {
        "long": {
            "standard": "שעון סיווה"
        }
    }

```

```

    }
  },
  "Tahiti": {
    "long": {
      "standard": "שעון טהיטי"
    }
  },
  "Taipei": {
    "long": {
      "generic": "שעון טאיפיי",
      "standard": "שעון טאיפיי (חורף)",
      "daylight": "שעון טאיפיי (קיץ)"
    }
  },
  "Tajikistan": {
    "long": {
      "standard": "שעון טג'יקיסטן"
    }
  },
  "Tokelau": {
    "long": {
      "standard": "שעון טוקלאו"
    }
  },
  "Tonga": {
    "long": {
      "generic": "שעון טונגה",
      "standard": "שעון טונגה (חורף)",
      "daylight": "שעון טונגה (קיץ)"
    }
  },
  "Truk": {
    "long": {
      "standard": "שעון צ'וק"
    }
  },
  "Turkmenistan": {
    "long": {
      "generic": "שעון טורקמניסטן",
      "standard": "שעון טורקמניסטן (חורף)",
      "daylight": "שעון טורקמניסטן (קיץ)"
    }
  },
  "Tuvalu": {
    "long": {
      "standard": "שעון טובאלו"
    }
  },
  "Uruguay": {
    "long": {
      "generic": "שעון אורוגוואי",
      "standard": "שעון אורוגוואי (חורף)",
      "daylight": "שעון אורוגוואי (קיץ)"
    }
  },
  "Uzbekistan": {
    "long": {

```

```

        "generic": "שעון אוזבקיסטן",
        "standard": "שעון אוזבקיסטן (חורף)",
        "daylight": "שעון אוזבקיסטן (קיץ)"
    },
},
"Vanuatu": {
    "long": {
        "generic": "שעון ונואטו",
        "standard": "שעון ונואטו (חורף)",
        "daylight": "שעון ונואטו (קיץ)"
    }
},
"Venezuela": {
    "long": {
        "standard": "שעון ונצואלה"
    }
},
"Vladivostok": {
    "long": {
        "generic": "שעון ולדיווסטוק",
        "standard": "שעון ולדיווסטוק (חורף)",
        "daylight": "שעון ולדיווסטוק (קיץ)"
    }
},
"Volgograd": {
    "long": {
        "generic": "שעון וולגוגרד",
        "standard": "שעון וולגוגרד (חורף)",
        "daylight": "שעון וולגוגרד (קיץ)"
    }
},
"Vostok": {
    "long": {
        "standard": "שעון ווסטוק"
    }
},
"Wake": {
    "long": {
        "standard": "שעון האי וייק"
    }
},
"Wallis": {
    "long": {
        "standard": "שעון וואליס ופוטונה"
    }
},
"Yakutsk": {
    "long": {
        "generic": "שעון יקוטסק",
        "standard": "שעון יקוטסק (חורף)",
        "daylight": "שעון יקוטסק (קיץ)"
    }
},
"Yekaterinburg": {
    "long": {
        "generic": "שעון יקטרינבורג",
        "standard": "שעון יקטרינבורג (חורף)",

```



```
"daylight": "שטון יקטרינבורג (קיץ)"
```

TIMEZONENAMES.JSX

```
{
  "main";
  {
    "he";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 13259 $",
          "_cldrVersion";
          "31.0.1";
        }
        "language";
        "he";
      }
    }
    "dates";
    {
      "timeZoneNames";
      {
        "hourFormat";
        "+HH:mm;-HH:mm",
        "gmtFormat";
        "GMT{0}",
        "gmtZeroFormat";
        "GMT",
        "regionFormat";
        "לעש {0}",
        "regionFormat-type-daylight";
        ")ק'ק} (0{ לעש",
        "regionFormat-type-standard";
        ")ק'ק} (0{ לעש",
        "fallbackFormat";
        "{1} ({0})",
        "zone";
        {
          "America";
          {
            "Adak";
            {
              "exemplarCity";
              "ק'ק";
            }
          }
        }
      }
    }
  }
}
```

```
"Anchorage";
{
  "exemplarCity";
  "אנקורג'";
}
"Anguilla";
{
  "exemplarCity";
  "אנגוויילה";
}
"Antigua";
{
  "exemplarCity";
  "אנטיואה";
}
"Araguaina";
{
  "exemplarCity";
  "אראגואינה";
}
"Argentina";
{
  "Rio_Gallegos";
  {
    "exemplarCity";
    "ריו גאליגוס";
  }
  "San_Juan";
  {
    "exemplarCity";
    "סן חואן";
  }
  "Ushuaia";
  {
    "exemplarCity";
    "אושואיה";
  }
  "La_Rioja";
  {
    "exemplarCity";
    "לה ריוחה";
  }
  "San_Luis";
  {
    "exemplarCity";
    "סן לואיס";
  }
  "Salta";
  {
    "exemplarCity";
    "סלטה";
  }
  "Tucuman";
  {
    "exemplarCity";
    "טוקומן";
  }
}
```

```
}
"Aruba";
{
  "exemplarCity";
  "ארובה";
}
"Asuncion";
{
  "exemplarCity";
  "אסונסיון";
}
"Bahia";
{
  "exemplarCity";
  "באהיה";
}
"Bahia_Banderas";
{
  "exemplarCity";
  "באהיה בנדרס";
}
"Barbados";
{
  "exemplarCity";
  "ברבדוס";
}
"Belem";
{
  "exemplarCity";
  "בלם";
}
"Belize";
{
  "exemplarCity";
  "בליז";
}
"Blanc-Sablon";
{
  "exemplarCity";
  "בלאן-סבלון";
}
"Boa_Vista";
{
  "exemplarCity";
  "בואה ויסטה";
}
"Bogota";
{
  "exemplarCity";
  "בוגוטה";
}
"Boise";
{
  "exemplarCity";
  "בוויסי";
}
"Buenos_Aires";
```

```
{
    "exemplarCity";
    "בואנוס איירס";
}
"Cambridge_Bay";
{
    "exemplarCity";
    "קיימברידג' ביי";
}
"Campo_Grande";
{
    "exemplarCity";
    "קמפּו גרנדה";
}
"Cancun";
{
    "exemplarCity";
    "קנקון";
}
"Caracas";
{
    "exemplarCity";
    "קראקס";
}
"Catamarca";
{
    "exemplarCity";
    "קטמרקה";
}
"Cayenne";
{
    "exemplarCity";
    "קאיייל";
}
"Cayman";
{
    "exemplarCity";
    "קיימן";
}
"Chicago";
{
    "exemplarCity";
    "שיקגו";
}
"Chihuahua";
{
    "exemplarCity";
    "צ'יוואוואה";
}
"Coral_Harbour";
{
    "exemplarCity";
    "אטיקוקל";
}
"Cordoba";
{
    "exemplarCity";
```

```
        "קורדובה";
    }
    "Costa_Rica";
    {
        "exemplarCity";
        "קוסטה ריקה";
    }
    "Creston";
    {
        "exemplarCity";
        "קרסטון";
    }
    "Cuiaba";
    {
        "exemplarCity";
        "קויאבה";
    }
    "Curacao";
    {
        "exemplarCity";
        "קוראטאו";
    }
    "Danmarkshavn";
    {
        "exemplarCity";
        "דנמרקסהוון";
    }
    "Dawson";
    {
        "exemplarCity";
        "דוסון";
    }
    "Dawson_Creek";
    {
        "exemplarCity";
        "דוסון קריק";
    }
    "Denver";
    {
        "exemplarCity";
        "דנוור";
    }
    "Detroit";
    {
        "exemplarCity";
        "דטרויט";
    }
    "Dominica";
    {
        "exemplarCity";
        "דומיניקה";
    }
    "Edmonton";
    {
        "exemplarCity";
        "אדמונטון";
    }
}
```

```
"Eirunepe";
{
  "exemplarCity";
  "אירונפי";
}
"El_Salvador";
{
  "exemplarCity";
  "אל סלבדור";
}
"Fort_Nelson";
{
  "exemplarCity";
  "פורט נלסון";
}
"Fortaleza";
{
  "exemplarCity";
  "פורטאלזה";
}
"Glace_Bay";
{
  "exemplarCity";
  "גלייס ביי";
}
"Godthab";
{
  "exemplarCity";
  "גואוק";
}
"Goose_Bay";
{
  "exemplarCity";
  "גוס ביי";
}
"Grand_Turk";
{
  "exemplarCity";
  "גרנד טורק";
}
"Grenada";
{
  "exemplarCity";
  "גרנדה";
}
"Guadeloupe";
{
  "exemplarCity";
  "גואדלופ";
}
"Guatemala";
{
  "exemplarCity";
  "גואטמלה";
}
"Guayaquil";
{
```

```

        "exemplarCity";
        "גואיאקיל";
    }
    "Guyana";
    {
        "exemplarCity";
        "גיאנה";
    }
    "Halifax";
    {
        "exemplarCity";
        "הליפקס";
    }
    "Havana";
    {
        "exemplarCity";
        "הוואנה";
    }
    "Hermosillo";
    {
        "exemplarCity";
        "הרמוסיו";
    }
    "Indiana";
    {
        "Vincennes";
        {
            "exemplarCity";
            "ווינסנס, אינדיאנה";
        }
        "Petersburg";
        {
            "exemplarCity";
            "פיטרסבורג, אינדיאנה";
        }
        "Tell_City";
        {
            "exemplarCity";
            "טל סיטי, אינדיאנה";
        }
        "Knox";
        {
            "exemplarCity";
            "נוקס, אינדיאנה";
        }
        "Winamac";
        {
            "exemplarCity";
            "ווינמאק, אינדיאנה";
        }
        "Marengo";
        {
            "exemplarCity";
            "מרנגו, אינדיאנה";
        }
        "Vevay";
        {

```

```

        "exemplarCity";
        "וינוויי, אינדיאנה";
    }
}
"Indianapolis";
{
    "exemplarCity";
    "אינדיאנפוליס";
}
"Inuvik";
{
    "exemplarCity";
    "אינוויק";
}
"Iqaluit";
{
    "exemplarCity";
    "אינקלוויט";
}
"Jamaica";
{
    "exemplarCity";
    "ג'מייקה";
}
"Jujuy";
{
    "exemplarCity";
    "ח'ח'ח";
}
"Juneau";
{
    "exemplarCity";
    "ג'ונ'ו";
}
"Kentucky";
{
    "Monticello";
    {
        "exemplarCity";
        "מונטיצ'לו, קנטאקי";
    }
}
"Kralendijk";
{
    "exemplarCity";
    "קרלנדייק";
}
"La_Paz";
{
    "exemplarCity";
    "לה פאס";
}
"Lima";
{
    "exemplarCity";
    "לימה";
}
}

```



```
"Los_Angeles";
{
  "exemplarCity";
  "לוס אנג'לס";
}
"Louisville";
{
  "exemplarCity";
  "לואיוויל";
}
"Lower_Princes";
{
  "exemplarCity";
  "לואוור פרינסס קוורטר";
}
"Maceio";
{
  "exemplarCity";
  "מסיאו";
}
"Managua";
{
  "exemplarCity";
  "מנגואה";
}
"Manaus";
{
  "exemplarCity";
  "מנאוס";
}
"Marigot";
{
  "exemplarCity";
  "מריגו";
}
"Martinique";
{
  "exemplarCity";
  "מרטיניק";
}
"Matamoros";
{
  "exemplarCity";
  "מטמורוס";
}
"Mazatlan";
{
  "exemplarCity";
  "מזטלן";
}
"Mendoza";
{
  "exemplarCity";
  "מנדוזה";
}
"Menominee";
{
```

```
        "exemplarCity";
        "מנוחייני";
    }
    "Merida";
    {
        "exemplarCity";
        "מרידה";
    }
    "Metlakatla";
    {
        "exemplarCity";
        "מטלקטלה";
    }
    "Mexico_City";
    {
        "exemplarCity";
        "מקסיקו סיטי";
    }
    "Miquelon";
    {
        "exemplarCity";
        "מיקלון";
    }
    "Moncton";
    {
        "exemplarCity";
        "מונקטון";
    }
    "Monterrey";
    {
        "exemplarCity";
        "מונטריי";
    }
    "Montevideo";
    {
        "exemplarCity";
        "מונטווידאו";
    }
    "Montserrat";
    {
        "exemplarCity";
        "מונטראט";
    }
    "Nassau";
    {
        "exemplarCity";
        "נסאו";
    }
    "New_York";
    {
        "exemplarCity";
        "ניו יורק";
    }
    "Nipigon";
    {
        "exemplarCity";
        "ניפיגון";
    }
```

```
}
"Nome";
{
  "exemplarCity";
  "נום";
}
"Noronha";
{
  "exemplarCity";
  "נורוניה";
}
"North_Dakota";
{
  "Beulah";
  {
    "exemplarCity";
    "ביולה, צפון דקוטה";
  }
  "New_Salem";
  {
    "exemplarCity";
    "ניו סיילם, צפון דקוטה";
  }
  "Center";
  {
    "exemplarCity";
    "סנטר, צפון דקוטה";
  }
}
"Ojinaga";
{
  "exemplarCity";
  "אוג'ינאגה";
}
"Panama";
{
  "exemplarCity";
  "פנמה";
}
"Pangnirtung";
{
  "exemplarCity";
  "פנגנירטונג";
}
"Paramaribo";
{
  "exemplarCity";
  "פרמריבו";
}
"Phoenix";
{
  "exemplarCity";
  "פיניקס";
}
"Port-au-Prince";
{
  "exemplarCity";
```

```
        "פורט או פראנס";
    }
    "Port_of_Spain";
    {
        "exemplarCity";
        "פורט אוף ספייַן";
    }
    "Porto_Velho";
    {
        "exemplarCity";
        "פורטו וליֹו";
    }
    "Puerto_Rico";
    {
        "exemplarCity";
        "פוארטו ריקו";
    }
    "Rainy_River";
    {
        "exemplarCity";
        "רייני ריבר";
    }
    "Rankin_Inlet";
    {
        "exemplarCity";
        "רנקין אינלט";
    }
    "Recife";
    {
        "exemplarCity";
        "רסיפה";
    }
    "Regina";
    {
        "exemplarCity";
        "רג'ינה";
    }
    "Resolute";
    {
        "exemplarCity";
        "רזולוט";
    }
    "Rio_Branco";
    {
        "exemplarCity";
        "ריו ברנקו";
    }
    "Santa_Isabel";
    {
        "exemplarCity";
        "סנטה איסבל";
    }
    "Santarem";
    {
        "exemplarCity";
        "סנטרם";
    }
}
```

```
"Santiago";
{
  "exemplarCity";
  "סנטיאגו";
}
"Santo_Domingo";
{
  "exemplarCity";
  "סנטו דומינגו";
}
"Sao_Paulo";
{
  "exemplarCity";
  "סאו פאולו";
}
"Scoresbysund";
{
  "exemplarCity";
  "סקורסביסונד";
}
"Sitka";
{
  "exemplarCity";
  "סיטקה";
}
"St_Barthelemy";
{
  "exemplarCity";
  "סנט ברתלמי";
}
"St_Johns";
{
  "exemplarCity";
  "סנט ג'ונס";
}
"St_Kitts";
{
  "exemplarCity";
  "סנט קיטס";
}
"St_Lucia";
{
  "exemplarCity";
  "סנט לוסיה";
}
"St_Thomas";
{
  "exemplarCity";
  "סנט תומאס";
}
"St_Vincent";
{
  "exemplarCity";
  "סנט וינסנט";
}
"Swift_Current";
{
```

```
        "exemplarCity";
        "סוויפט קרנט";
    }
    "Tegucigalpa";
    {
        "exemplarCity";
        "טגוסיגלפה";
    }
    "Thule";
    {
        "exemplarCity";
        "תולה";
    }
    "Thunder_Bay";
    {
        "exemplarCity";
        "ת'אנדר ביי";
    }
    "Tijuana";
    {
        "exemplarCity";
        "טיחואנה";
    }
    "Toronto";
    {
        "exemplarCity";
        "טורונטו";
    }
    "Tortola";
    {
        "exemplarCity";
        "טורטולה";
    }
    "Vancouver";
    {
        "exemplarCity";
        "ונקובר";
    }
    "Whitehorse";
    {
        "exemplarCity";
        "ווייטהורס";
    }
    "Winnipeg";
    {
        "exemplarCity";
        "וויניפג";
    }
    "Yakutat";
    {
        "exemplarCity";
        "יקוטאט";
    }
    "Yellowknife";
    {
        "exemplarCity";
        "ילן ייף";
    }
}
```

```

    }
  }
  "Atlantic";
  {
    "Azores";
    {
      "exemplarCity";
      "האיים האזוריים";
    }
    "Bermuda";
    {
      "exemplarCity";
      "ברמודה";
    }
    "Canary";
    {
      "exemplarCity";
      "האיים הקנריים";
    }
    "Cape_Verde";
    {
      "exemplarCity";
      "כף ורדה";
    }
    "Faeroe";
    {
      "exemplarCity";
      "פארו";
    }
    "Madeira";
    {
      "exemplarCity";
      "מדירה";
    }
    "Reykjavik";
    {
      "exemplarCity";
      "רייקיאויק";
    }
    "South_Georgia";
    {
      "exemplarCity";
      "דרום ג'ורג'יה";
    }
    "St_Helena";
    {
      "exemplarCity";
      "סנט הלנה";
    }
    "Stanley";
    {
      "exemplarCity";
      "סטנלי";
    }
  }
  "Europe";
  {

```

```
"Amsterdam";
{
  "exemplarCity";
  "אמסטרדם";
}
"Andorra";
{
  "exemplarCity";
  "אנדורה";
}
"Astrakhan";
{
  "exemplarCity";
  "אסטרחן";
}
"Athens";
{
  "exemplarCity";
  "אתונה";
}
"Belgrade";
{
  "exemplarCity";
  "בלגרד";
}
"Berlin";
{
  "exemplarCity";
  "ברלין";
}
"Bratislava";
{
  "exemplarCity";
  "ברטיסלבה";
}
"Brussels";
{
  "exemplarCity";
  "בריסל";
}
"Bucharest";
{
  "exemplarCity";
  "בוקרשט";
}
"Budapest";
{
  "exemplarCity";
  "בודפשט";
}
"Busingen";
{
  "exemplarCity";
  "ביזינגן";
}
"Chisinau";
{
```



```
        "exemplarCity";
        "קִישִׁינְב";
    }
    "Copenhagen";
    {
        "exemplarCity";
        "קופנהגן";
    }
    "Dublin";
    {
        "long";
        {
            "daylight";
            "שעון קיץ אירלנד";
        }
        "exemplarCity";
        "דבלין";
    }
    "Gibraltar";
    {
        "exemplarCity";
        "גיברלטר";
    }
    "Guernsey";
    {
        "exemplarCity";
        "גרנזי";
    }
    "Helsinki";
    {
        "exemplarCity";
        "הלסינקי";
    }
    "Isle_of_Man";
    {
        "exemplarCity";
        "האי מאן";
    }
    "Istanbul";
    {
        "exemplarCity";
        "איסטנבול";
    }
    "Jersey";
    {
        "exemplarCity";
        "ג'רזי";
    }
    "Kaliningrad";
    {
        "exemplarCity";
        "קלינינגרד";
    }
    "Kiev";
    {
        "exemplarCity";
        "קייב";
    }
```

```
}
"Kirov";
{
  "exemplarCity";
  "קִירוֹב";
}
"Lisbon";
{
  "exemplarCity";
  "לִיסְבוֹן";
}
"Ljubljana";
{
  "exemplarCity";
  "לוֹבְלִיאָנָה";
}
"London";
{
  "long";
  {
    "daylight";
    "שְׁעוֹן קִיץ בְּרִיטְנִיָּה";
  }
  "exemplarCity";
  "לוֹנְדוֹן";
}
"Luxembourg";
{
  "exemplarCity";
  "לוֹקְסֶמְבוּרְג";
}
"Madrid";
{
  "exemplarCity";
  "מָדְרִיד";
}
"Malta";
{
  "exemplarCity";
  "מַלְטָה";
}
"Mariehamn";
{
  "exemplarCity";
  "מַרִּיהָאֵמֶן";
}
"Minsk";
{
  "exemplarCity";
  "מִינְסְק";
}
"Monaco";
{
  "exemplarCity";
  "מוֹנָקוֹ";
}
"Moscow";
```

```
{
  "exemplarCity";
  "מוסקבה";
}
"Oslo";
{
  "exemplarCity";
  "אוסלו";
}
"Paris";
{
  "exemplarCity";
  "פריז";
}
"Podgorica";
{
  "exemplarCity";
  "פודגוריצה";
}
"Prague";
{
  "exemplarCity";
  "פראג";
}
"Riga";
{
  "exemplarCity";
  "ריגה";
}
"Rome";
{
  "exemplarCity";
  "רומא";
}
"Samara";
{
  "exemplarCity";
  "סמרה";
}
"San_Marino";
{
  "exemplarCity";
  "סן מרינו";
}
"Sarajevo";
{
  "exemplarCity";
  "סרייבו";
}
"Simferopol";
{
  "exemplarCity";
  "סימפרופול";
}
"Skopje";
{
  "exemplarCity";
```

```
        "סְקוֹפִיָּה";
    }
    "Sofia";
    {
        "exemplarCity";
        "סוֹפִיָּה";
    }
    "Stockholm";
    {
        "exemplarCity";
        "שְׁטוֹקְהוֹלְם";
    }
    "Tallinn";
    {
        "exemplarCity";
        "טַאָלִין";
    }
    "Tirane";
    {
        "exemplarCity";
        "טִירֵנָה";
    }
    "Ulyanovsk";
    {
        "exemplarCity";
        "אֻלְיָאנוֹבְסְק";
    }
    "Uzhgorod";
    {
        "exemplarCity";
        "אֻזְהוֹרֹוד";
    }
    "Vaduz";
    {
        "exemplarCity";
        "וַאדוּץ";
    }
    "Vatican";
    {
        "exemplarCity";
        "הוֹוֵתִיקָן";
    }
    "Vienna";
    {
        "exemplarCity";
        "וִיֵנָה";
    }
    "Vilnius";
    {
        "exemplarCity";
        "וִילְנָה";
    }
    "Volgograd";
    {
        "exemplarCity";
        "וּוֹלְגוֹגְרַד";
    }
}
```

```
"Warsaw";
{
  "exemplarCity";
  "ורשה";
}
"Zagreb";
{
  "exemplarCity";
  "זאגרב";
}
"Zaporozhye";
{
  "exemplarCity";
  "זפורוז'יה";
}
"Zurich";
{
  "exemplarCity";
  "ציריך";
}
}
"Africa";
{
  "Abidjan";
  {
    "exemplarCity";
    "אביג'אן";
  }
  "Accra";
  {
    "exemplarCity";
    "אקרה";
  }
  "Addis_Ababa";
  {
    "exemplarCity";
    "אדיס אבבה";
  }
  "Algiers";
  {
    "exemplarCity";
    "אלג'יר";
  }
  "Asmera";
  {
    "exemplarCity";
    "אסמרה";
  }
  "Bamako";
  {
    "exemplarCity";
    "במאקו";
  }
  "Bangui";
  {
    "exemplarCity";
    "בנגווי";
  }
}
```

```
}
"Banjul";
{
  "exemplarCity";
  "בנג'ול";
}
"Bissau";
{
  "exemplarCity";
  "ביסאו";
}
"Blantyre";
{
  "exemplarCity";
  "בלנטיר";
}
"Brazzaville";
{
  "exemplarCity";
  "ברזוויל";
}
"Bujumbura";
{
  "exemplarCity";
  "בוג'ומבורה";
}
"Cairo";
{
  "exemplarCity";
  "קהיר";
}
"Casablanca";
{
  "exemplarCity";
  "קזבלנקה";
}
"Ceuta";
{
  "exemplarCity";
  "סאוטה";
}
"Conakry";
{
  "exemplarCity";
  "קונאקרי";
}
"Dakar";
{
  "exemplarCity";
  "דקאר";
}
"Dar_es_Salaam";
{
  "exemplarCity";
  "דאר-א-סלאם";
}
"Djibouti";
```

```
{
  "exemplarCity";
  "ג'יבוטי";
}
"Douala";
{
  "exemplarCity";
  "דואלה";
}
"El_Aaiun";
{
  "exemplarCity";
  "אל עיון";
}
"Freetown";
{
  "exemplarCity";
  "פריטאון";
}
"Gaborone";
{
  "exemplarCity";
  "גבורונה";
}
"Harare";
{
  "exemplarCity";
  "הרארה";
}
"Johannesburg";
{
  "exemplarCity";
  "יוהנסבורג";
}
"Juba";
{
  "exemplarCity";
  "ג'ובה";
}
"Kampala";
{
  "exemplarCity";
  "קמפאלה";
}
"Khartoum";
{
  "exemplarCity";
  "חרטום";
}
"Kigali";
{
  "exemplarCity";
  "קיגלי";
}
"Kinshasa";
{
  "exemplarCity";
```

```
        "קנישסה";
    }
    "Lagos";
    {
        "exemplarCity";
        "לגוס";
    }
    "Libreville";
    {
        "exemplarCity";
        "ליברוויל";
    }
    "Lome";
    {
        "exemplarCity";
        "לומה";
    }
    "Luanda";
    {
        "exemplarCity";
        "לואאנדה";
    }
    "Lubumbashi";
    {
        "exemplarCity";
        "לובומבאשי";
    }
    "Lusaka";
    {
        "exemplarCity";
        "לוסקה";
    }
    "Malabo";
    {
        "exemplarCity";
        "מלבו";
    }
    "Maputo";
    {
        "exemplarCity";
        "מאפוטו";
    }
    "Maseru";
    {
        "exemplarCity";
        "מסרו";
    }
    "Mbabane";
    {
        "exemplarCity";
        "אמבאבאנה";
    }
    "Mogadishu";
    {
        "exemplarCity";
        "מוגדישו";
    }
}
```



```
"Monrovia";
{
  "exemplarCity";
  "מונרוביה";
}
"Nairobi";
{
  "exemplarCity";
  "ניירובי";
}
"Ndjamena";
{
  "exemplarCity";
  "נג'מנה";
}
"Niamey";
{
  "exemplarCity";
  "ניאמי";
}
"Nouakchott";
{
  "exemplarCity";
  "נואקצ'וט";
}
"Ouagadougou";
{
  "exemplarCity";
  "וואגאדוגו";
}
"Porto-Novo";
{
  "exemplarCity";
  "פורטו נובו";
}
"Sao_Tome";
{
  "exemplarCity";
  "סאו טומה";
}
"Tripoli";
{
  "exemplarCity";
  "טריפולי";
}
"Tunis";
{
  "exemplarCity";
  "תוניס";
}
"Windhoek";
{
  "exemplarCity";
  "ווינדהוק";
}
}
"Asia";
```

```
{
  "Aden";
  {
    "exemplarCity";
    "עדן";
  }
  "Almaty";
  {
    "exemplarCity";
    "אלמאטי";
  }
  "Amman";
  {
    "exemplarCity";
    "עמאן";
  }
  "Anadyr";
  {
    "exemplarCity";
    "אנדיר";
  }
  "Aqtau";
  {
    "exemplarCity";
    "אקטאו";
  }
  "Aqtobe";
  {
    "exemplarCity";
    "אקטובה";
  }
  "Ashgabat";
  {
    "exemplarCity";
    "אשגבט";
  }
  "Baghdad";
  {
    "exemplarCity";
    "בגדד";
  }
  "Bahrain";
  {
    "exemplarCity";
    "בחרין";
  }
  "Baku";
  {
    "exemplarCity";
    "באקו";
  }
  "Bangkok";
  {
    "exemplarCity";
    "בנגקוק";
  }
  "Barnaul";
```

```
{
  "exemplarCity";
  "ברנאול";
}
"Beirut";
{
  "exemplarCity";
  "ביירות";
}
"Bishkek";
{
  "exemplarCity";
  "בישקק";
}
"Brunei";
{
  "exemplarCity";
  "ברוניי";
}
"Calcutta";
{
  "exemplarCity";
  "קולקטה";
}
"Chita";
{
  "exemplarCity";
  "צ'יטה";
}
"Choibalsan";
{
  "exemplarCity";
  "צ'ויבלסן";
}
"Colombo";
{
  "exemplarCity";
  "קולומבו";
}
"Damascus";
{
  "exemplarCity";
  "דמשק";
}
"Dhaka";
{
  "exemplarCity";
  "דאקה";
}
"Dili";
{
  "exemplarCity";
  "דילי";
}
"Dubai";
{
  "exemplarCity";
```

```

        "דובאי";
    }
    "Dushanbe";
    {
        "exemplarCity";
        "דושנבה";
    }
    "Gaza";
    {
        "exemplarCity";
        "עזה";
    }
    "Hebron";
    {
        "exemplarCity";
        "חברון";
    }
    "Hong_Kong";
    {
        "exemplarCity";
        "הונג קונג";
    }
    "Hovd";
    {
        "exemplarCity";
        "חובד";
    }
    "Irkutsk";
    {
        "exemplarCity";
        "אירקוטסק";
    }
    "Jakarta";
    {
        "exemplarCity";
        "ג'קרטה";
    }
    "Jayapura";
    {
        "exemplarCity";
        "ג'איאפורה";
    }
    "Jerusalem";
    {
        "exemplarCity";
        "ירושלים";
    }
    "Kabul";
    {
        "exemplarCity";
        "קאבול";
    }
    "Kamchatka";
    {
        "exemplarCity";
        "קמצ'טקה";
    }
}

```

```
"Karachi";
{
  "exemplarCity";
  "קראצ'י";
}
"Katmandu";
{
  "exemplarCity";
  "קטמנדו";
}
"Khandyga";
{
  "exemplarCity";
  "חנדיגה";
}
"Krasnoyarsk";
{
  "exemplarCity";
  "קרסנויארסק";
}
"Kuala_Lumpur";
{
  "exemplarCity";
  "קואלה לומפור";
}
"Kuching";
{
  "exemplarCity";
  "קוצ'ינג";
}
"Kuwait";
{
  "exemplarCity";
  "כווית";
}
"Macau";
{
  "exemplarCity";
  "מקאו";
}
"Magadan";
{
  "exemplarCity";
  "מגדל";
}
"Makassar";
{
  "exemplarCity";
  "מאקאסאר";
}
"Manila";
{
  "exemplarCity";
  "מנילה";
}
"Muscat";
{
```

```

        "exemplarCity";
        "מוסקט";
    }
    "Nicosia";
    {
        "exemplarCity";
        "ניקוסיה";
    }
    "Novokuznetsk";
    {
        "exemplarCity";
        "נובוקוזנטסק";
    }
    "Novosibirsk";
    {
        "exemplarCity";
        "נובוסירסק";
    }
    "Omsk";
    {
        "exemplarCity";
        "אומסק";
    }
    "Oral";
    {
        "exemplarCity";
        "אורל";
    }
    "Phnom_Penh";
    {
        "exemplarCity";
        "פנום פן";
    }
    "Pontianak";
    {
        "exemplarCity";
        "פונטיאנק";
    }
    "Pyongyang";
    {
        "exemplarCity";
        "פיונגיאנג";
    }
    "Qatar";
    {
        "exemplarCity";
        "קטאר";
    }
    "Qyzylorda";
    {
        "exemplarCity";
        "קזיילורדה";
    }
    "Rangoon";
    {
        "exemplarCity";
        "רנגון";
    }

```

```
}
"Riyadh";
{
  "exemplarCity";
  "ריאד";
}
"Saigon";
{
  "exemplarCity";
  "הו צ'י מין סיטי";
}
"Sakhalin";
{
  "exemplarCity";
  "סחלין";
}
"Samarkand";
{
  "exemplarCity";
  "סמרקנד";
}
"Seoul";
{
  "exemplarCity";
  "סיאול";
}
"Shanghai";
{
  "exemplarCity";
  "שנחאי";
}
"Singapore";
{
  "exemplarCity";
  "סינגפור";
}
"Srednekolymsk";
{
  "exemplarCity";
  "סרדנייקולימסק";
}
"Taipei";
{
  "exemplarCity";
  "טאיפיי";
}
"Tashkent";
{
  "exemplarCity";
  "טשקנט";
}
"Tbilisi";
{
  "exemplarCity";
  "טביליסי";
}
"Tehran";
```

```
{
    "exemplarCity";
    "טהרן";
}
"Thimphu";
{
    "exemplarCity";
    "טהימפהו";
}
"Tokyo";
{
    "exemplarCity";
    "טוקיו";
}
"Tomsk";
{
    "exemplarCity";
    "טומסק";
}
"Ulaanbaatar";
{
    "exemplarCity";
    "אולאאנבטאר";
}
"Urumqi";
{
    "exemplarCity";
    "אורומקי";
}
"Ust-Nera";
{
    "exemplarCity";
    "אוסט-נרה";
}
"Vientiane";
{
    "exemplarCity";
    "האנוי";
}
"Vladivostok";
{
    "exemplarCity";
    "ולדיוווסטוק";
}
"Yakutsk";
{
    "exemplarCity";
    "יקוטסק";
}
"Yekaterinburg";
{
    "exemplarCity";
    "יקטרינבורג";
}
"Yerevan";
{
    "exemplarCity";
```



```

        "ירושלם";
    }
}
"Indian";
{
    "Antananarivo";
    {
        "exemplarCity";
        "אנטננריבו";
    }
    "Chagos";
    {
        "exemplarCity";
        "צ'אגוס";
    }
    "Christmas";
    {
        "exemplarCity";
        "האי כריסטמס";
    }
    "Cocos";
    {
        "exemplarCity";
        "קוקוס";
    }
    "Comoro";
    {
        "exemplarCity";
        "קומורו";
    }
    "Kerguelen";
    {
        "exemplarCity";
        "קרגוולן";
    }
    "Mahe";
    {
        "exemplarCity";
        "מהא";
    }
    "Maldives";
    {
        "exemplarCity";
        "האיים המלדיביים";
    }
    "Mauritius";
    {
        "exemplarCity";
        "מאוריציוס";
    }
    "Mayotte";
    {
        "exemplarCity";
        "מאיוט";
    }
    "Reunion";
    {

```

```
        "exemplarCity";
        "ראוניון";
    }
}
"Australia";
{
    "Adelaide";
    {
        "exemplarCity";
        "אדלייד";
    }
    "Brisbane";
    {
        "exemplarCity";
        "בריסביין";
    }
    "Broken_Hill";
    {
        "exemplarCity";
        "ברוקן היל";
    }
    "Currie";
    {
        "exemplarCity";
        "קרי";
    }
    "Darwin";
    {
        "exemplarCity";
        "דרווין";
    }
    "Eucla";
    {
        "exemplarCity";
        "יוקלה";
    }
    "Hobart";
    {
        "exemplarCity";
        "הוברט";
    }
    "Lindeman";
    {
        "exemplarCity";
        "לינדמן";
    }
    "Lord_Howe";
    {
        "exemplarCity";
        "אי הלורד האו";
    }
    "Melbourne";
    {
        "exemplarCity";
        "מלבורן";
    }
    "Perth";
```

```

        {
            "exemplarCity";
            "פרת'";
        }
        "Sydney";
        {
            "exemplarCity";
            "סידני";
        }
    }
    "Pacific";
    {
        "Apia";
        {
            "exemplarCity";
            "אפיה";
        }
        "Auckland";
        {
            "exemplarCity";
            "אוקלנד";
        }
        "Bougainville";
        {
            "exemplarCity";
            "בוגנוויל";
        }
        "Chatham";
        {
            "exemplarCity";
            "צ'אטהאם";
        }
        "Easter";
        {
            "exemplarCity";
            "א'י הפסחא";
        }
        "Efate";
        {
            "exemplarCity";
            "אפטח";
        }
        "Enderbury";
        {
            "exemplarCity";
            "אנדרבורי";
        }
        "Fakaofo";
        {
            "exemplarCity";
            "פקאופו";
        }
        "Fiji";
        {
            "exemplarCity";
            "פיג'י";
        }
    }

```

```
"Funafuti";
{
  "exemplarCity";
  "פונפוטִי";
}
"Galapagos";
{
  "exemplarCity";
  "גלפאגוס";
}
"Gambier";
{
  "exemplarCity";
  "איי גמבייה";
}
"Guadalcanal";
{
  "exemplarCity";
  "גוודלקנאל";
}
"Guam";
{
  "exemplarCity";
  "גואם";
}
"Honolulu";
{
  "exemplarCity";
  "הונולוֹלוֹ";
}
"Johnston";
{
  "exemplarCity";
  "ג'ונסטון";
}
"Kiritimati";
{
  "exemplarCity";
  "קיריטימאטי";
}
"Kosrae";
{
  "exemplarCity";
  "קוסרה";
}
"Kwajalein";
{
  "exemplarCity";
  "קוואג'ליין";
}
"Majuro";
{
  "exemplarCity";
  "מאג'ורו";
}
"Marquesas";
{
```

```
        "exemplarCity";
        "איי מרקז";
    }
    "Midway";
    {
        "exemplarCity";
        "מידוויי";
    }
    "Nauru";
    {
        "exemplarCity";
        "נאורו";
    }
    "Niue";
    {
        "exemplarCity";
        "ניואה";
    }
    "Norfolk";
    {
        "exemplarCity";
        "נורפוק";
    }
    "Noumea";
    {
        "exemplarCity";
        "נומאה";
    }
    "Pago_Pago";
    {
        "exemplarCity";
        "פאגו פאגו";
    }
    "Palau";
    {
        "exemplarCity";
        "פלאו";
    }
    "Pitcairn";
    {
        "exemplarCity";
        "פיטקרן";
    }
    "Ponape";
    {
        "exemplarCity";
        "פונפיי";
    }
    "Port_Moresby";
    {
        "exemplarCity";
        "פורט מורסבי";
    }
    "Rarotonga";
    {
        "exemplarCity";
        "רארוטונגה";
    }
```

```
}
    "Saipan";
    {
        "exemplarCity";
        "סאַיפאַן";
    }
    "Tahiti";
    {
        "exemplarCity";
        "טהיטי";
    }
    "Tarawa";
    {
        "exemplarCity";
        "טאַראווה";
    }
    "Tongatapu";
    {
        "exemplarCity";
        "טונגטאַפּו";
    }
    "Truk";
    {
        "exemplarCity";
        "צ'וק";
    }
    "Wake";
    {
        "exemplarCity";
        "ווייק";
    }
    "Wallis";
    {
        "exemplarCity";
        "וואליס";
    }
}
"Arctic";
{
    "Longyearbyen";
    {
        "exemplarCity";
        "לונגיירבין";
    }
}
"Antarctica";
{
    "Casey";
    {
        "exemplarCity";
        "קאַסיי";
    }
    "Davis";
    {
        "exemplarCity";
        "דייוויס";
    }
}
```

```

        "DumontDUrville";
        {
            "exemplarCity";
            "דומון ד'אורוויל";
        }
        "Macquarie";
        {
            "exemplarCity";
            "מקרי";
        }
        "Mawson";
        {
            "exemplarCity";
            "מוסון";
        }
        "McMurdo";
        {
            "exemplarCity";
            "מק-מרדו";
        }
        "Palmer";
        {
            "exemplarCity";
            "פאלמר";
        }
        "Rothera";
        {
            "exemplarCity";
            "רות'רה";
        }
        "Syowa";
        {
            "exemplarCity";
            "סייוואה";
        }
        "Troll";
        {
            "exemplarCity";
            "טרול";
        }
        "Vostok";
        {
            "exemplarCity";
            "ווסטוק";
        }
    }
    "Etc";
    {
        "GMT";
        {
            "exemplarCity";
            "GMT";
        }
        "GMT1";
        {
            "exemplarCity";
            "GMT+1";
        }
    }

```

```
}
"GMT10";
{
  "exemplarCity";
  "GMT+10";
}
"GMT11";
{
  "exemplarCity";
  "GMT+11";
}
"GMT12";
{
  "exemplarCity";
  "GMT+12";
}
"GMT2";
{
  "exemplarCity";
  "GMT+2";
}
"GMT3";
{
  "exemplarCity";
  "GMT+3";
}
"GMT4";
{
  "exemplarCity";
  "GMT+4";
}
"GMT5";
{
  "exemplarCity";
  "GMT+5";
}
"GMT6";
{
  "exemplarCity";
  "GMT+6";
}
"GMT7";
{
  "exemplarCity";
  "GMT+7";
}
"GMT8";
{
  "exemplarCity";
  "GMT+8";
}
"GMT9";
{
  "exemplarCity";
  "GMT+9";
}
"GMT-1";
```



```
{
    "exemplarCity";
    "GMT-1";
}
"GMT-10";
{
    "exemplarCity";
    "GMT-10";
}
"GMT-11";
{
    "exemplarCity";
    "GMT-11";
}
"GMT-12";
{
    "exemplarCity";
    "GMT-12";
}
"GMT-13";
{
    "exemplarCity";
    "GMT-13";
}
"GMT-14";
{
    "exemplarCity";
    "GMT-14";
}
"GMT-2";
{
    "exemplarCity";
    "GMT-2";
}
"GMT-3";
{
    "exemplarCity";
    "GMT-3";
}
"GMT-4";
{
    "exemplarCity";
    "GMT-4";
}
"GMT-5";
{
    "exemplarCity";
    "GMT-5";
}
"GMT-6";
{
    "exemplarCity";
    "GMT-6";
}
"GMT-7";
{
    "exemplarCity";
```

```

        "GMT-7";
    }
    "GMT-8";
    {
        "exemplarCity";
        "GMT-8";
    }
    "GMT-9";
    {
        "exemplarCity";
        "GMT-9";
    }
    "UTC";
    {
        "long";
        {
            "standard";
            "זמן אוניברסלי מתואם";
        }
        "short";
        {
            "standard";
            "UTC";
        }
        "exemplarCity";
        "UTC";
    }
    "Unknown";
    {
        "exemplarCity";
        "עיר לא ידועה";
    }
    }
    "metazone";
    {
        "Afghanistan";
        {
            "long";
            {
                "standard";
                "שעון אפגניסטן";
            }
        }
        "Africa_Central";
        {
            "long";
            {
                "standard";
                "שעון מרכז אפריקה";
            }
        }
        "Africa_Eastern";
        {
            "long";
            {
                "standard";
            }
        }
    }

```

```

        "שעון מזרח אפריקה";
    }
}
"Africa_Southern";
{
    "long";
    {
        "standard";
        "שעון דרום אפריקה";
    }
}
"Africa_Western";
{
    "long";
    {
        "generic";
        "שעון מערב אפריקה",
        "standard";
        "שעון מערב אפריקה (חורף)",
        "daylight";
        "שעון מערב אפריקה (קיץ)";
    }
}
}
"Alaska";
{
    "long";
    {
        "generic";
        "שעון אלסקה",
        "standard";
        "שעון אלסקה (חורף)",
        "daylight";
        "שעון אלסקה (קיץ)";
    }
}
}
"Amazon";
{
    "long";
    {
        "generic";
        "שעון אמזונס",
        "standard";
        "שעון אמזונס (חורף)",
        "daylight";
        "שעון אמזונס (קיץ)";
    }
}
}
"America_Central";
{
    "long";
    {
        "generic";
        "שעון מרכז ארה"ב",
        "standard";
        "שעון מרכז ארה"ב (חורף)",
        "daylight";
        "שעון מרכז ארה"ב (קיץ)";
    }
}

```

```
    }  
  }  
  "America_Eastern";  
  {  
    "long";  
    {  
      "generic";  
      "שעון החוף המזרחי",  
      "standard";  
      "שעון החוף המזרחי (חורף)",  
      "daylight";  
      "שעון החוף המזרחי (קיץ)";  
    }  
  }  
  "America_Mountain";  
  {  
    "long";  
    {  
      "generic";  
      "שעון אזור ההרים בארה"ב",  
      "standard";  
      "שעון אזור ההרים בארה"ב (חורף)",  
      "daylight";  
      "שעון אזור ההרים בארה"ב (קיץ)";  
    }  
  }  
  "America_Pacific";  
  {  
    "long";  
    {  
      "generic";  
      "שעון מערב ארה"ב",  
      "standard";  
      "שעון מערב ארה"ב (חורף)",  
      "daylight";  
      "שעון מערב ארה"ב (קיץ)";  
    }  
  }  
  "Anadyr";  
  {  
    "long";  
    {  
      "generic";  
      "שעון אנדיר",  
      "standard";  
      "שעון רגיל אנדיר",  
      "daylight";  
      "שעון קיץ אנדיר";  
    }  
  }  
  "Apia";  
  {  
    "long";  
    {  
      "generic";  
      "שעון אפיה",  
      "standard";  
    }  
  }
```

```
        "שעון אפיה (חורף)",
        "daylight";
        "שעון אפיה (קיץ)";
    }
}
"Arabian";
{
    "long";
    {
        "generic";
        "שעון חצי האי ערב",
        "standard";
        "שעון חצי האי ערב (חורף)",
        "daylight";
        "שעון חצי האי ערב (קיץ)";
    }
}
"Argentina";
{
    "long";
    {
        "generic";
        "שעון ארגנטינה",
        "standard";
        "שעון ארגנטינה (חורף)",
        "daylight";
        "שעון ארגנטינה (קיץ)";
    }
}
"Argentina_Western";
{
    "long";
    {
        "generic";
        "שעון מערב ארגנטינה",
        "standard";
        "שעון מערב ארגנטינה (חורף)",
        "daylight";
        "שעון מערב ארגנטינה (קיץ)";
    }
}
"Armenia";
{
    "long";
    {
        "generic";
        "שעון ארמניה",
        "standard";
        "שעון ארמניה (חורף)",
        "daylight";
        "שעון ארמניה (קיץ)";
    }
}
"Atlantic";
{
    "long";
    {
```

```
        "generic";
        "שעון האוקיינוס האטלנטי",
        "standard";
        "שעון האוקיינוס האטלנטי (חורף)",
        "daylight";
        "שעון האוקיינוס האטלנטי (קיץ)";
    }
}
"Australia_Central";
{
    "long";
    {
        "generic";
        "שעון מרכז אוסטרליה",
        "standard";
        "שעון מרכז אוסטרליה (חורף)",
        "daylight";
        "שעון מרכז אוסטרליה (קיץ)";
    }
}
"Australia_CentralWestern";
{
    "long";
    {
        "generic";
        "שעון מרכז-מערב אוסטרליה",
        "standard";
        "שעון מרכז-מערב אוסטרליה (חורף)",
        "daylight";
        "שעון מרכז-מערב אוסטרליה (קיץ)";
    }
}
"Australia_Eastern";
{
    "long";
    {
        "generic";
        "שעון מזרח אוסטרליה",
        "standard";
        "שעון מזרח אוסטרליה (חורף)",
        "daylight";
        "שעון מזרח אוסטרליה (קיץ)";
    }
}
"Australia_Western";
{
    "long";
    {
        "generic";
        "שעון מערב אוסטרליה",
        "standard";
        "שעון מערב אוסטרליה (חורף)",
        "daylight";
        "שעון מערב אוסטרליה (קיץ)";
    }
}
"Azerbaijan";
```

```
{
  "long";
  {
    "generic";
    "שעון אזרבייג'אן",
    "standard";
    "שעון אזרבייג'אן (חורף)",
    "daylight";
    "שעון אזרבייג'אן (קיץ)";
  }
}
"Azores";
{
  "long";
  {
    "generic";
    "שעון האיים האזוריים",
    "standard";
    "שעון האיים האזוריים (חורף)",
    "daylight";
    "שעון האיים האזוריים (קיץ)";
  }
}
"Bangladesh";
{
  "long";
  {
    "generic";
    "שעון בנגלדש",
    "standard";
    "שעון בנגלדש (חורף)",
    "daylight";
    "שעון בנגלדש (קיץ)";
  }
}
"Bhutan";
{
  "long";
  {
    "standard";
    "שעון בהוטן";
  }
}
"Bolivia";
{
  "long";
  {
    "standard";
    "שעון בוליביה";
  }
}
"Brasilia";
{
  "long";
  {
    "generic";
    "שעון ברזיליה",
```

```
        "standard";
        "שעון ברזיליה (חורף)",
        "daylight";
        "שעון ברזיליה (קיץ)";
    }
}
"Brunei";
{
    "long";
    {
        "standard";
        "שעון ברוניי דארוסלאם";
    }
}
"Cape_Verde";
{
    "long";
    {
        "generic";
        "שעון כף ורדה",
        "standard";
        "שעון כף ורדה (חורף)",
        "daylight";
        "שעון כף ורדה (קיץ)";
    }
}
"Chamorro";
{
    "long";
    {
        "standard";
        "שעון צ'אמורו";
    }
}
"Chatham";
{
    "long";
    {
        "generic";
        "שעון צ'טהאם",
        "standard";
        "שעון צ'טהאם (חורף)",
        "daylight";
        "שעון צ'טהאם (קיץ)";
    }
}
"Chile";
{
    "long";
    {
        "generic";
        "שעון צ'ילה",
        "standard";
        "שעון צ'ילה (חורף)",
        "daylight";
        "שעון צ'ילה (קיץ)";
    }
}
```



```
}
"China";
{
  "long";
  {
    "generic";
    "שעון סין",
    "standard";
    "שעון סין (חורף)",
    "daylight";
    "שעון סין (קיץ)";
  }
}
"Choibalsan";
{
  "long";
  {
    "generic";
    "שעון צ'ויבלסן",
    "standard";
    "שעון צ'ויבלסן (חורף)",
    "daylight";
    "שעון צ'ויבלסן (קיץ)";
  }
}
"Christmas";
{
  "long";
  {
    "standard";
    "שעון האי כריסטמס";
  }
}
"Cocos";
{
  "long";
  {
    "standard";
    "שעון איי קוקוס";
  }
}
"Colombia";
{
  "long";
  {
    "generic";
    "שעון קולומביה",
    "standard";
    "שעון קולומביה (חורף)",
    "daylight";
    "שעון קולומביה (קיץ)";
  }
}
"Cook";
{
  "long";
  {
```

```

        "generic";
        "שעון איי קוק",
        "standard";
        "שעון איי קוק (חורף)",
        "daylight";
        "שעון איי קוק (מחצית הקיץ)";
    }
}
"Cuba";
{
    "long";
    {
        "generic";
        "שעון קובה",
        "standard";
        "שעון קובה (חורף)",
        "daylight";
        "שעון קובה (קיץ)";
    }
}
"Davis";
{
    "long";
    {
        "standard";
        "שעון דייוויס";
    }
}
"DumontDUrville";
{
    "long";
    {
        "standard";
        "שעון דומון ד'אורוויל";
    }
}
"East_Timor";
{
    "long";
    {
        "standard";
        "שעון מזרח טימור";
    }
}
"Easter";
{
    "long";
    {
        "generic";
        "שעון אי הפסחא",
        "standard";
        "שעון אי הפסחא (חורף)",
        "daylight";
        "שעון אי הפסחא (קיץ)";
    }
}
"Ecuador";

```

```

    {
      "long";
      {
        "standard";
        "שעון אקוודור";
      }
    }
    "Europe_Central";
    {
      "long";
      {
        "generic";
        "שעון מרכז אירופה",
        "standard";
        "שעון מרכז אירופה (חורף)",
        "daylight";
        "שעון מרכז אירופה (קיץ)";
      }
    }
    "Europe_Eastern";
    {
      "long";
      {
        "generic";
        "שעון מזרח אירופה",
        "standard";
        "שעון מזרח אירופה (חורף)",
        "daylight";
        "שעון מזרח אירופה (קיץ)";
      }
    }
    "Europe_Further_Eastern";
    {
      "long";
      {
        "standard";
        "שעון מינסק";
      }
    }
    "Europe_Western";
    {
      "long";
      {
        "generic";
        "שעון מערב אירופה",
        "standard";
        "שעון מערב אירופה (חורף)",
        "daylight";
        "שעון מערב אירופה (קיץ)";
      }
    }
    "Falkland";
    {
      "long";
      {
        "generic";
        "שעון איי פוקלנד",

```

```

        "standard";
        "שעון איי פוקלנד (חורף)",
        "daylight";
        "שעון איי פוקלנד (קיץ)";
    }
}
"Fiji";
{
    "long";
    {
        "generic";
        "שעון פיג'י",
        "standard";
        "שעון פיג'י (חורף)",
        "daylight";
        "שעון פיג'י (קיץ)";
    }
}
"French_Guiana";
{
    "long";
    {
        "standard";
        "שעון גיאנה הצרפתית";
    }
}
"French_Southern";
{
    "long";
    {
        "standard";
        "שעון הארצות הדרומיות והאנטארקטיות של צרפת";
    }
}
"Galapagos";
{
    "long";
    {
        "standard";
        "שעון איי גלאפגוס";
    }
}
"Gambier";
{
    "long";
    {
        "standard";
        "שעון איי גמבייה";
    }
}
"Georgia";
{
    "long";
    {
        "generic";
        "שעון גאורגיה",
        "standard";
    }
}

```

```

        "שעון גאורגיה (חורף)",
        "daylight";
        "שעון גאורגיה (קיץ)";
    }
}
"Gilbert_Islands";
{
    "long";
    {
        "standard";
        "שעון איי גילברט";
    }
}
"GMT";
{
    "long";
    {
        "standard";
        "שעון גריניץ'";
    }
}
"Greenland_Eastern";
{
    "long";
    {
        "generic";
        "שעון מזרח גרינלנד",
        "standard";
        "שעון מזרח גרינלנד (חורף)",
        "daylight";
        "שעון מזרח גרינלנד (קיץ)";
    }
}
"Greenland_Western";
{
    "long";
    {
        "generic";
        "שעון מערב גרינלנד",
        "standard";
        "שעון מערב גרינלנד (חורף)",
        "daylight";
        "שעון מערב גרינלנד (קיץ)";
    }
}
"Gulf";
{
    "long";
    {
        "standard";
        "שעון מדינות המפרץ";
    }
}
"Guyana";
{
    "long";
    {

```

```
        "standard";
        "שעון גיאנה";
    }
}
"Hawaii_Aleutian";
{
    "long";
    {
        "generic";
        "שעון האיים האלאוטיים הוואי",
        "standard";
        "שעון האיים האלאוטיים הוואי (חורף)",
        "daylight";
        "שעון האיים האלאוטיים הוואי (קיץ)";
    }
}
"Hong_Kong";
{
    "long";
    {
        "generic";
        "שעון הונג קונג",
        "standard";
        "שעון הונג קונג (חורף)",
        "daylight";
        "שעון הונג קונג (קיץ)";
    }
}
"Hovd";
{
    "long";
    {
        "generic";
        "שעון חובד",
        "standard";
        "שעון חובד (חורף)",
        "daylight";
        "שעון חובד (קיץ)";
    }
}
"India";
{
    "long";
    {
        "standard";
        "שעון הודו";
    }
}
"Indian_Ocean";
{
    "long";
    {
        "standard";
        "שעון האוקיינוס ההודי";
    }
}
"Indochina";
```

```

{
  "long";
  {
    "standard";
    "שעון הודו-סין";
  }
}
"Indonesia_Central";
{
  "long";
  {
    "standard";
    "שעון מרכז אינדונזיה";
  }
}
"Indonesia_Eastern";
{
  "long";
  {
    "standard";
    "שעון מזרח אינדונזיה";
  }
}
"Indonesia_Western";
{
  "long";
  {
    "standard";
    "שעון מערב אינדונזיה";
  }
}
"Iran";
{
  "long";
  {
    "generic";
    "שעון איראן",
    "standard";
    "שעון איראן (חורף)",
    "daylight";
    "שעון איראן (קיץ)";
  }
}
"Irkutsk";
{
  "long";
  {
    "generic";
    "שעון אירקוטסק",
    "standard";
    "שעון אירקוטסק (חורף)",
    "daylight";
    "שעון אירקוטסק (קיץ)";
  }
}
"Israel";
{

```

```
        "long";
        {
            "generic";
            "שעון ישראל",
            "standard";
            "שעון ישראל (חורף)",
            "daylight";
            "שעון ישראל (קיץ)";
        }
    }
    "Japan";
    {
        "long";
        {
            "generic";
            "שעון יפן",
            "standard";
            "שעון יפן (חורף)",
            "daylight";
            "שעון יפן (קיץ)";
        }
    }
    "Kamchatka";
    {
        "long";
        {
            "generic";
            "שעון פטרופלובסק-קמצ'טסקי",
            "standard";
            "שעון רגיל פטרופלובסק-קמצ'טסקי",
            "daylight";
            "שעון קיץ פטרופלובסק-קמצ'טסקי";
        }
    }
    "Kazakhstan_Eastern";
    {
        "long";
        {
            "standard";
            "שעון מזרח קזחסטן";
        }
    }
    "Kazakhstan_Western";
    {
        "long";
        {
            "standard";
            "שעון מערב קזחסטן";
        }
    }
    "Korea";
    {
        "long";
        {
            "generic";
            "שעון קוריאה",
            "standard";
```



```
        "שעון קוריאה (חורף)",
        "daylight";
        "שעון קוריאה (קיץ)";
    }
}
"Kosrae";
{
    "long";
    {
        "standard";
        "שעון קוסראה";
    }
}
"Krasnoyarsk";
{
    "long";
    {
        "generic";
        "שעון קרסנויארסק",
        "standard";
        "שעון קרסנויארסק (חורף)",
        "daylight";
        "שעון קרסנויארסק (קיץ)";
    }
}
"Kyrgystan";
{
    "long";
    {
        "standard";
        "שעון קירגיזסטן";
    }
}
"Line_Islands";
{
    "long";
    {
        "standard";
        "שעון איי ליין";
    }
}
"Lord_Howe";
{
    "long";
    {
        "generic";
        "שעון אי הלורד האו",
        "standard";
        "שעון אי הלורד האו (חורף)",
        "daylight";
        "שעון אי הלורד האו (קיץ)";
    }
}
"Macau";
{
    "long";
    {
```

```

        "generic";
        "שעון מקאו",
        "standard";
        "שעון חורף מקאו",
        "daylight";
        "שעון קיץ מקאו";
    }
}
"Macquarie";
{
    "long";
    {
        "standard";
        "שעון מקווארי";
    }
}
"Magadan";
{
    "long";
    {
        "generic";
        "שעון מגדן",
        "standard";
        "שעון מגדן (חורף)",
        "daylight";
        "שעון מגדן (קיץ)";
    }
}
"Malaysia";
{
    "long";
    {
        "standard";
        "שעון מלזיה";
    }
}
"Maldives";
{
    "long";
    {
        "standard";
        "שעון המלדיביים";
    }
}
"Marquesas";
{
    "long";
    {
        "standard";
        "שעון מרקז";
    }
}
"Marshall_Islands";
{
    "long";
    {
        "standard";
    }
}

```

```

        "שעון אייר מרשל";
    }
}
"Mauritius";
{
    "long";
    {
        "generic";
        "שעון מאוריציוס",
        "standard";
        "שעון מאוריציוס (חורף)",
        "daylight";
        "שעון מאוריציוס (קיץ)";
    }
}
"Mawson";
{
    "long";
    {
        "standard";
        "שעון מאוסון";
    }
}
"Mexico_Northwest";
{
    "long";
    {
        "generic";
        "שעון צפון-מערב מקסיקו",
        "standard";
        "שעון צפון-מערב מקסיקו (חורף)",
        "daylight";
        "שעון צפון-מערב מקסיקו (קיץ)";
    }
}
"Mexico_Pacific";
{
    "long";
    {
        "generic";
        "שעון מערב מקסיקו",
        "standard";
        "שעון מערב מקסיקו (חורף)",
        "daylight";
        "שעון מערב מקסיקו (קיץ)";
    }
}
"Mongolia";
{
    "long";
    {
        "generic";
        "שעון אולן בטור",
        "standard";
        "שעון אולן בטור (חורף)",
        "daylight";
        "שעון אולן בטור (קיץ)";
    }
}

```

```
    }  
  }  
  "Moscow";  
  {  
    "long";  
    {  
      "generic";  
      "שעון מוסקבה",  
      "standard";  
      "שעון מוסקבה (חורף)",  
      "daylight";  
      "שעון מוסקבה (קיץ)";  
    }  
  }  
  "Myanmar";  
  {  
    "long";  
    {  
      "standard";  
      "שעון מיאנמר";  
    }  
  }  
  "Nauru";  
  {  
    "long";  
    {  
      "standard";  
      "שעון נאורו";  
    }  
  }  
  "Nepal";  
  {  
    "long";  
    {  
      "standard";  
      "שעון נפאל";  
    }  
  }  
  "New_Caledonia";  
  {  
    "long";  
    {  
      "generic";  
      "שעון קלדוניה החדשה",  
      "standard";  
      "שעון קלדוניה החדשה (חורף)",  
      "daylight";  
      "שעון קלדוניה החדשה (קיץ)";  
    }  
  }  
  "New_Zealand";  
  {  
    "long";  
    {  
      "generic";  
      "שעון ניו זילנד",  
      "standard";  
    }  
  }
```

```

        "שעון ניו זילנד (חורף)",
        "daylight";
        "שעון ניו זילנד (קיץ)";
    }
}
"Newfoundland";
{
    "long";
    {
        "generic";
        "שעון ניופאונדלנד",
        "standard";
        "שעון ניופאונדלנד (חורף)",
        "daylight";
        "שעון ניופאונדלנד (קיץ)";
    }
}
"Niue";
{
    "long";
    {
        "standard";
        "שעון ניואה";
    }
}
"Norfolk";
{
    "long";
    {
        "standard";
        "שעון האי נורפוק";
    }
}
"Noronha";
{
    "long";
    {
        "generic";
        "שעון פרננדו די נורוניה",
        "standard";
        "שעון פרננדו די נורוניה (חורף)",
        "daylight";
        "שעון פרננדו די נורוניה (קיץ)";
    }
}
"Novosibirsk";
{
    "long";
    {
        "generic";
        "שעון נובוסיבירסק",
        "standard";
        "שעון נובוסיבירסק (חורף)",
        "daylight";
        "שעון נובוסיבירסק (קיץ)";
    }
}
}

```

```
"Omsk";
{
  "long";
  {
    "generic";
    "שעון אומסק",
    "standard";
    "שעון אומסק (חורף)",
    "daylight";
    "שעון אומסק (קיץ)";
  }
}
"Pakistan";
{
  "long";
  {
    "generic";
    "שעון פקיסטן",
    "standard";
    "שעון פקיסטן (חורף)",
    "daylight";
    "שעון פקיסטן (קיץ)";
  }
}
"Palau";
{
  "long";
  {
    "standard";
    "שעון פלאו";
  }
}
"Papua_New_Guinea";
{
  "long";
  {
    "standard";
    "שעון פפואה גיניאה החדשה";
  }
}
"Paraguay";
{
  "long";
  {
    "generic";
    "שעון פרגוואי",
    "standard";
    "שעון פרגוואי (חורף)",
    "daylight";
    "שעון פרגוואי (קיץ)";
  }
}
"Peru";
{
  "long";
  {
    "generic";
```

```

        "שעון פרו",
        "standard";
        "שעון פרו (חורף)",
        "daylight";
        "שעון פרו (קיץ)";
    }
}
"Philippines";
{
    "long";
    {
        "generic";
        "שעון הפיליפינים",
        "standard";
        "שעון הפיליפינים (חורף)",
        "daylight";
        "שעון הפיליפינים (קיץ)";
    }
}
"Phoenix_Islands";
{
    "long";
    {
        "standard";
        "שעון איי פניקס";
    }
}
"Pierre_Miquelon";
{
    "long";
    {
        "generic";
        "שעון סנט פייר ומיקלון",
        "standard";
        "שעון סנט פייר ומיקלון (חורף)",
        "daylight";
        "שעון סנט פייר ומיקלון (קיץ)";
    }
}
"Pitcairn";
{
    "long";
    {
        "standard";
        "שעון פיטקרן";
    }
}
"Ponape";
{
    "long";
    {
        "standard";
        "שעון פונאפי";
    }
}
"Pyongyang";
{

```

```
        "long";
        {
            "standard";
            "שעון פיונגיאנג";
        }
    }
    "Reunion";
    {
        "long";
        {
            "standard";
            "שעון ראוניון";
        }
    }
    "Rothera";
    {
        "long";
        {
            "standard";
            "שעון רות'רה";
        }
    }
    "Sakhalin";
    {
        "long";
        {
            "generic";
            "שעון סחלין",
            "standard";
            "שעון סחלין (חורף)",
            "daylight";
            "שעון סחלין (קיץ)";
        }
    }
    "Samara";
    {
        "long";
        {
            "generic";
            "שעון סמרה",
            "standard";
            "שעון רגיל סמרה",
            "daylight";
            "שעון קיץ סמרה";
        }
    }
    "Samoa";
    {
        "long";
        {
            "generic";
            "שעון סמואה",
            "standard";
            "שעון סמואה (חורף)",
            "daylight";
            "שעון סמואה (קיץ)";
        }
    }
}
```



```
}
"Seychelles";
{
  "long";
  {
    "standard";
    "שעון איי סיישל";
  }
}
"Singapore";
{
  "long";
  {
    "standard";
    "שעון סינגפור";
  }
}
"Solomon";
{
  "long";
  {
    "standard";
    "שעון איי שלמה";
  }
}
"South_Georgia";
{
  "long";
  {
    "standard";
    "שעון דרום ג'ורג'יה";
  }
}
"Suriname";
{
  "long";
  {
    "standard";
    "שעון סורינאם";
  }
}
"Syowa";
{
  "long";
  {
    "standard";
    "שעון סייווה";
  }
}
"Tahiti";
{
  "long";
  {
    "standard";
    "שעון טהיטי";
  }
}
}
```

```
"Taipei";
{
  "long";
  {
    "generic";
    "שעון טאיפיי",
    "standard";
    "שעון טאיפיי (חורף)",
    "daylight";
    "שעון טאיפיי (קיץ)";
  }
}
"Tajikistan";
{
  "long";
  {
    "standard";
    "שעון טג'יקיסטן";
  }
}
"Tokelau";
{
  "long";
  {
    "standard";
    "שעון טוקלאו";
  }
}
"Tonga";
{
  "long";
  {
    "generic";
    "שעון טונגה",
    "standard";
    "שעון טונגה (חורף)",
    "daylight";
    "שעון טונגה (קיץ)";
  }
}
"Truk";
{
  "long";
  {
    "standard";
    "שעון צ'וק";
  }
}
"Turkmenistan";
{
  "long";
  {
    "generic";
    "שעון טורקמניסטן",
    "standard";
    "שעון טורקמניסטן (חורף)",
    "daylight";
  }
}
```

```

        "שעון טורקמניסטן (קיץ)";
    }
}
"Tuvalu";
{
    "long";
    {
        "standard";
        "שעון טובאלו";
    }
}
"Uruguay";
{
    "long";
    {
        "generic";
        "שעון אורוגוואי",
        "standard";
        "שעון אורוגוואי (חורף)",
        "daylight";
        "שעון אורוגוואי (קיץ)";
    }
}
"Uzbekistan";
{
    "long";
    {
        "generic";
        "שעון אוזבקיסטן",
        "standard";
        "שעון אוזבקיסטן (חורף)",
        "daylight";
        "שעון אוזבקיסטן (קיץ)";
    }
}
"Vanuatu";
{
    "long";
    {
        "generic";
        "שעון ונואטו",
        "standard";
        "שעון ונואטו (חורף)",
        "daylight";
        "שעון ונואטו (קיץ)";
    }
}
"Venezuela";
{
    "long";
    {
        "standard";
        "שעון ונצואלה";
    }
}
"Vladivostok";
{

```

```

        "long";
        {
            "generic";
            "שעון ולדיווסטוק",
            "standard";
            "שעון ולדיווסטוק (חורף)",
            "daylight";
            "שעון ולדיווסטוק (קיץ)";
        }
    }
    "Volgograd";
    {
        "long";
        {
            "generic";
            "שעון וולגוגרד",
            "standard";
            "שעון וולגוגרד (חורף)",
            "daylight";
            "שעון וולגוגרד (קיץ)";
        }
    }
    "Vostok";
    {
        "long";
        {
            "standard";
            "שעון ווסטוק";
        }
    }
    "Wake";
    {
        "long";
        {
            "standard";
            "שעון האי וייק";
        }
    }
    "Wallis";
    {
        "long";
        {
            "standard";
            "שעון וואליס ופוטונה";
        }
    }
    "Yakutsk";
    {
        "long";
        {
            "generic";
            "שעון יקוטסק",
            "standard";
            "שעון יקוטסק (חורף)",
            "daylight";
            "שעון יקוטסק (קיץ)";
        }
    }

```

```
}
    "Yekaterinburg";
    {
        "long";
        {
            "generic";
            "שעון יקטרינבורג",
            "standard";
            "(שעון יקטרינבורג) חורף",
            "daylight";
            "(שעון יקטרינבורג) קיץ";
        }
    }
}
}
```

CA-GREGORIAN.JSON

```
{
  "main": {
    "he": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31.0.1"
        },
        "language": "he"
      },
      "dates": {
        "calendars": {
          "gregorian": {
            "months": {
              "format": {
                "abbreviated": {
                  "1": "ינו',",
                  "2": "פבר',",
                  "3": "מרץ",
                  "4": "אפר',",
                  "5": "מאי",
                  "6": "יוני",
                  "7": "יולי",
                  "8": "אוג",
                  "9": "ספט",
                  "10": "אוק',",
                  "11": "נוב",
                  "12": "דצמ'"
                },
                "narrow": {
                  "1": "1",
                  "2": "2",
```

```
    "3": "3",
    "4": "4",
    "5": "5",
    "6": "6",
    "7": "7",
    "8": "8",
    "9": "9",
    "10": "10",
    "11": "11",
    "12": "12"
  },
  "wide": {
    "1": "ינואר",
    "2": "פברואר",
    "3": "מרץ",
    "4": "אפריל",
    "5": "מאי",
    "6": "יוני",
    "7": "יולי",
    "8": "אוגוסט",
    "9": "ספטמבר",
    "10": "אוקטובר",
    "11": "נובמבר",
    "12": "דצמבר"
  }
},
"stand-alone": {
  "abbreviated": {
    "1": "ינו'",
    "2": "פבר'",
    "3": "מרץ",
    "4": "אפר'",
    "5": "מאי",
    "6": "יוני",
    "7": "יולי",
    "8": "אוג'",
    "9": "ספט'",
    "10": "אוק'",
    "11": "נוב'",
    "12": "דצמ'"
  },
  "narrow": {
    "1": "1",
    "2": "2",
    "3": "3",
    "4": "4",
    "5": "5",
    "6": "6",
    "7": "7",
    "8": "8",
    "9": "9",
    "10": "10",
    "11": "11",
    "12": "12"
  },
  "wide": {
    "1": "ינואר",
```

```

        "2": "פברואר",
        "3": "מרץ",
        "4": "אפריל",
        "5": "מאי",
        "6": "יוני",
        "7": "יולי",
        "8": "אוגוסט",
        "9": "ספטמבר",
        "10": "אוקטובר",
        "11": "נובמבר",
        "12": "דצמבר"
    }
},
"days": {
    "format": {
        "abbreviated": {
            "sun": "א' יום",
            "mon": "ב' יום",
            "tue": "ג' יום",
            "wed": "ד' יום",
            "thu": "ה' יום",
            "fri": "ו' יום",
            "sat": "שבת"
        },
        "narrow": {
            "sun": "א",
            "mon": "ב",
            "tue": "ג",
            "wed": "ד",
            "thu": "ה",
            "fri": "ו",
            "sat": "ש"
        },
        "short": {
            "sun": "א'",
            "mon": "ב'",
            "tue": "ג'",
            "wed": "ד'",
            "thu": "ה'",
            "fri": "ו'",
            "sat": "ש'"
        },
        "wide": {
            "sun": "יום ראשון",
            "mon": "יום שני",
            "tue": "יום שלישי",
            "wed": "יום רביעי",
            "thu": "יום חמישי",
            "fri": "יום שישי",
            "sat": "יום שבת"
        }
    },
    "stand-alone": {
        "abbreviated": {
            "sun": "א' יום",
            "mon": "ב' יום",

```

```

        "tue": "יום ג'",
        "wed": "יום ד'",
        "thu": "יום ה'",
        "fri": "יום ו'",
        "sat": "שבת"
    },
    "narrow": {
        "sun": "א'",
        "mon": "ב'",
        "tue": "ג'",
        "wed": "ד'",
        "thu": "ה'",
        "fri": "ו'",
        "sat": "ש'"
    },
    "short": {
        "sun": "א",
        "mon": "ב",
        "tue": "ג",
        "wed": "ד",
        "thu": "ה",
        "fri": "ו",
        "sat": "ש"
    },
    "wide": {
        "sun": "יום ראשון",
        "mon": "יום שני",
        "tue": "יום שלישי",
        "wed": "יום רביעי",
        "thu": "יום חמישי",
        "fri": "יום שישי",
        "sat": "יום שבת"
    }
},
"quarters": {
    "format": {
        "abbreviated": {
            "1": "Q1",
            "2": "Q2",
            "3": "Q3",
            "4": "Q4"
        },
        "narrow": {
            "1": "1",
            "2": "2",
            "3": "3",
            "4": "4"
        },
        "wide": {
            "1": "1 רבעון",
            "2": "2 רבעון",
            "3": "3 רבעון",
            "4": "4 רבעון"
        }
    }
},
"stand-alone": {

```



```

    "abbreviated": {
      "1": "Q1",
      "2": "Q2",
      "3": "Q3",
      "4": "Q4"
    },
    "narrow": {
      "1": "1",
      "2": "2",
      "3": "3",
      "4": "4"
    },
    "wide": {
      "1": "1 רבעון",
      "2": "2 רבעון",
      "3": "3 רבעון",
      "4": "4 רבעון"
    }
  },
  "dayPeriods": {
    "format": {
      "abbreviated": {
        "midnight": "חצות",
        "am": "לפנה״צ",
        "pm": "אחה״צ",
        "morning1": "בוקר",
        "afternoon1": "צהריים",
        "afternoon2": "אחר הצהריים",
        "evening1": "ערב",
        "night1": "לילה",
        "night2": "לפנות בוקר"
      },
      "narrow": {
        "midnight": "חצות",
        "am": "לפנה״צ",
        "pm": "אחה״צ",
        "morning1": "בוקר",
        "afternoon1": "צהריים",
        "afternoon2": "אחר הצהריים",
        "evening1": "ערב",
        "night1": "לילה",
        "night2": "לפנות בוקר"
      },
      "wide": {
        "midnight": "חצות",
        "am": "לפנה״צ",
        "pm": "אחה״צ",
        "morning1": "בוקר",
        "afternoon1": "צהריים",
        "afternoon2": "אחר הצהריים",
        "evening1": "ערב",
        "night1": "לילה",
        "night2": "לפנות בוקר"
      }
    },
    "stand-alone": {

```

```

    "abbreviated": {
      "midnight": "חצות",
      "am": "לפנה"צ",
      "pm": "אחה"צ",
      "morning1": "בוקר",
      "afternoon1": "צהריים",
      "afternoon2": "אחר הצהריים",
      "evening1": "ערב",
      "night1": "לילה",
      "night2": "לפנות בוקר"
    },
    "narrow": {
      "midnight": "חצות",
      "am": "לפנה"צ",
      "pm": "אחה"צ",
      "morning1": "בוקר",
      "afternoon1": "צהריים",
      "afternoon2": "אחר הצהריים",
      "evening1": "ערב",
      "night1": "לילה",
      "night2": "לפנות בוקר"
    },
    "wide": {
      "midnight": "חצות",
      "am": "לפנה"צ",
      "pm": "אחה"צ",
      "morning1": "בוקר",
      "afternoon1": "צהריים",
      "afternoon2": "אחר הצהריים",
      "evening1": "ערב",
      "night1": "לילה",
      "night2": "לפנות בוקר"
    }
  },
  "eras": {
    "eraNames": {
      "0": "לפני הספירה",
      "0-alt-variant": "לפנה"ס",
      "1": "לספירה",
      "1-alt-variant": "CE"
    },
    "eraAbbr": {
      "0": "לפנה"ס",
      "0-alt-variant": "BCE",
      "1": "לספירה",
      "1-alt-variant": "CE"
    },
    "eraNarrow": {
      "0": "לפנה"ס",
      "0-alt-variant": "BCE",
      "1": "לספירה",
      "1-alt-variant": "CE"
    }
  },
  "dateFormats": {
    "full": "EEEE, d MMMM y",

```

```

    "long": "d מMMMM y",
    "medium": "d מMMM y",
    "short": "d.M.y"
  },
  "timeFormats": {
    "full": "H:mm:ss zzzz",
    "long": "H:mm:ss z",
    "medium": "H:mm:ss",
    "short": "H:mm"
  },
  "dateTimeFormats": {
    "full": "{1} בשעה {0}",
    "long": "{1} בשעה {0}",
    "medium": "{1}, {0}",
    "short": "{1}, {0}",
    "availableFormats": {
      "d": "d",
      "E": "ccc",
      "Ed": "E ה-d",
      "Ehm": "E h:mm a",
      "EHm": "E H:mm",
      "Ehms": "E h:mm:ss a",
      "EHms": "E H:mm:ss",
      "Gy": "y G",
      "GyMMM": "MMM y G",
      "GyMMMd": "d מMMM y G",
      "GyMMMED": "E, d מMMM y G",
      "h": "h a",
      "H": "H",
      "hm": "h:mm a",
      "Hm": "H:mm",
      "hms": "h:mm:ss a",
      "Hms": "H:mm:ss",
      "hmsv": "h:mm:ss a v",
      "Hmsv": "HH:mm:ss v",
      "hmv": "h:mm a v",
      "Hmv": "HH:mm v",
      "M": "L",
      "Md": "d.M",
      "MEd": "E, d.M",
      "MMM": "LLL",
      "MMMd": "d מMMM",
      "MMMED": "E, d מMMM",
      "MMMMd": "d מMMMM",
      "MMMMW-count-one": "שבוע W מMMM",
      "MMMMW-count-two": "שבוע W מMMM",
      "MMMMW-count-many": "שבוע W מMMM",
      "MMMMW-count-other": "שבוע W מMMM",
      "ms": "mm:ss",
      "y": "y",
      "yM": "M.y",
      "yMd": "d.M.y",
      "yMEd": "E, d.M.y",
      "yMM": "M.y",
      "yMMM": "MMM y",
      "yMMMd": "d מMMM y",
      "yMMMED": "E, d מMMM y",

```

```

    "yMMMM": "MMMM y",
    "yQQQ": "QQQ y",
    "yQQQQ": "QQQQ y",
    "yw-count-one": "y בשנת w שבוטע",
    "yw-count-two": "y בשנת w שבוטע",
    "yw-count-many": "y בשנת w שבוטע",
    "yw-count-other": "y בשנת w שבוטע"
  },
  "appendItems": {
    "Day": "{0} ({2}: {1})",
    "Day-Of-Week": "{0} {1}",
    "Era": "{1} {0}",
    "Hour": "{0} ({2}: {1})",
    "Minute": "{0} ({2}: {1})",
    "Month": "{0} ({2}: {1})",
    "Quarter": "{0} ({2}: {1})",
    "Second": "{0} ({2}: {1})",
    "Timezone": "{0} {1}",
    "Week": "{0} ({2}: {1})",
    "Year": "{1} {0}"
  },
  "intervalFormats": {
    "intervalFormatFallback": "{0} - {1}",
    "d": {
      "d": "d-d"
    },
    "h": {
      "a": "h a - h a",
      "h": "h-h a"
    },
    "H": {
      "H": "H-H"
    },
    "hm": {
      "a": "h:mm a - h:mm a",
      "h": "h:mm-h:mm a",
      "m": "h:mm-h:mm a"
    },
    "Hm": {
      "H": "H:mm-H:mm",
      "m": "H:mm-H:mm"
    },
    "hmv": {
      "a": "h:mm a - h:mm a v",
      "h": "h:mm-h:mm a v",
      "m": "h:mm-h:mm a v"
    },
    "Hmv": {
      "H": "H:mm-H:mm v",
      "m": "H:mm-H:mm v"
    },
    "hv": {
      "a": "h a - h a v",
      "h": "h-h a v"
    },
    "Hv": {
      "H": "H-H v"
    }
  }

```

```

    },
    "M": {
      "M": "M-M"
    },
    "Md": {
      "d": "d.M-d.M",
      "M": "d.M-d.M"
    },
    "MEd": {
      "d": "EEEE d.M-EEEE d.M",
      "M": "EEEE d.M - EEEE d.M"
    },
    "MMM": {
      "M": "MMM-MMM"
    },
    "MMMd": {
      "d": "d-d 1MMM",
      "M": "d 1MMM-d 1MMM"
    },
    "MMMEd": {
      "d": "EEEE, d 1MMM - EEEE, d 1MMM",
      "M": "EEEE, d 1MMM - EEEE, d 1MMM"
    },
    "MMMM": {
      "M": "LLLL-LLLL"
    },
    "Y": {
      "Y": "Y-Y"
    },
    "YM": {
      "M": "M.Y-M.Y",
      "Y": "M.Y-M.Y"
    },
    "YMd": {
      "d": "dd.M.y - dd.M.y",
      "M": "d.M.y - d.M.y",
      "Y": "d.M.y - d.M.y"
    },
    "YMEd": {
      "d": "EEEE d.M.y - EEEE d.M.y",
      "M": "EEEE d.M.y - EEEE d.M.y",
      "Y": "EEEE d.M.y - EEEE d.M.y"
    },
    "YMMM": {
      "M": "MMM-MMM y",
      "Y": "MMM y - MMM y"
    },
    "YMMMd": {
      "d": "d-d 1MMM y",
      "M": "d MMM - d MMM y",
      "Y": "d MMM y - d MMM y"
    },
    "YMMMEd": {
      "d": "EEEE d MMM - EEEE d MMM y",
      "M": "EEEE d MMM - EEEE d MMM y",
      "Y": "EEEE d MMM y - EEEE d MMM y"
    },
  },

```

```
"YMMMMM": {  
    "M": "MMMM-MMMM Y",  
    "Y": "MMMM Y-MMMM Y"  
}  
  
}  
  
}  
  
}  
  
}  
  
}  
  
}
```

CA-GREGORIAN.JSX

```
{
    "main";
    {
        "he";
        {
            "identity";
            {
                "version";
                {
                    "_number";
                    "$Revision: 13259 $",
                    "_cldrVersion";
                    "31.0.1";
                }
                "language";
                "he";
            }
        }
        "dates";
        {
            "calendars";
            {
                "gregorian";
                {
                    "months";
                    {
                        "format";
                        {
                            "abbreviated";
                            {
                                "1";
                                "ינוני",
                                "2";
                                "פבר",
                                "3";
                                "מרץ",
                                "4";
                                "אפר",
                                "5";
                                "מאי",
                                "6";
                                "יוני",
                                "7";
                                "יולי",
                                "8";
                                "אוגוסט",
                                "9";
                                "ספט",
                                "10";
                                "אוקט",
                                "11";
                                "נוב",
                                "12";
                                "דצמבר";
                            }
                        }
                    }
                }
            }
        }
    }
}
```

```

        "7";
        "יולי",
        "8";
        "אוג",
        "9";
        "ספט",
        "10";
        "אוק",
        "11";
        "נוב",
        "12";
        "דצמ";
    }
    "narrow";
    {
        "1";
        "1",
        "2";
        "2",
        "3";
        "3",
        "4";
        "4",
        "5";
        "5",
        "6";
        "6",
        "7";
        "7",
        "8";
        "8",
        "9";
        "9",
        "10";
        "10",
        "11";
        "11",
        "12";
        "12";
    }
    "wide";
    {
        "1";
        "ינואר",
        "2";
        "פברואר",
        "3";
        "מרץ",
        "4";
        "אפריל",
        "5";
        "מאי",
        "6";
        "יוני",
        "7";
        "יולי",
        "8";
    }

```

```

        "אוגוסט",
        "9";
        "ספטמבר",
        "10";
        "אוקטובר",
        "11";
        "נובמבר",
        "12";
        "דצמבר";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
        "ינו'",
        "2";
        "פבר'",
        "3";
        "מרץ",
        "4";
        "אפר'",
        "5";
        "מאי",
        "6";
        "יוני",
        "7";
        "יולי",
        "8";
        "אוג'",
        "9";
        "ספט'",
        "10";
        "אוק'",
        "11";
        "נוב'",
        "12";
        "דצמ'";
    }
}
"narrow";
{
    "1";
    "1",
    "2";
    "2",
    "3";
    "3",
    "4";
    "4",
    "5";
    "5",
    "6";
    "6",
    "7";
    "7",
    "8";

```



```

        "8",
        "9";
        "9",
        "10";
        "10",
        "11";
        "11",
        "12";
        "12";
    }
    "wide";
    {
        "1";
        "ינואר",
        "2";
        "פברואר",
        "3";
        "מרץ",
        "4";
        "אפריל",
        "5";
        "מאי",
        "6";
        "יוני",
        "7";
        "יולי",
        "8";
        "אוגוסט",
        "9";
        "ספטמבר",
        "10";
        "אוקטובר",
        "11";
        "נובמבר",
        "12";
        "דצמבר";
    }
}
}
"days";
{
    "format";
    {
        "abbreviated";
        {
            "sun";
            "יום א'",
            "mon";
            "יום ב'",
            "tue";
            "יום ג'",
            "wed";
            "יום ד'",
            "thu";
            "יום ה'",
            "fri";
            "יום ו'";
        }
    }
}

```

```

        "sat";
        "שבת";
    }
    "narrow";
    {
        "sun";
        "א'",
        "mon";
        "ב'",
        "tue";
        "ג'",
        "wed";
        "ד'",
        "thu";
        "ה'",
        "fri";
        "ו'",
        "sat";
        "ש";
    }
    "short";
    {
        "sun";
        "א'",
        "mon";
        "ב'",
        "tue";
        "ג'",
        "wed";
        "ד'",
        "thu";
        "ה'",
        "fri";
        "ו'",
        "sat";
        "ש";
    }
    "wide";
    {
        "sun";
        "יום ראשון",
        "mon";
        "יום שני",
        "tue";
        "יום שלישי",
        "wed";
        "יום רביעי",
        "thu";
        "יום חמישי",
        "fri";
        "יום שישי",
        "sat";
        "יום שבת";
    }
    }
    "stand-alone";
    {

```

```
"abbreviated";
{
  "sun";
  "י' א'",
  "mon";
  "ב' י'",
  "tue";
  "ג' י'",
  "wed";
  "ד' י'",
  "thu";
  "ה' י'",
  "fri";
  "ו' י'",
  "sat";
  "שבת";
}
"narrow";
{
  "sun";
  "א'",
  "mon";
  "ב'",
  "tue";
  "ג'",
  "wed";
  "ד'",
  "thu";
  "ה'",
  "fri";
  "ו'",
  "sat";
  "ש'";
}
"short";
{
  "sun";
  "א'",
  "mon";
  "ב'",
  "tue";
  "ג'",
  "wed";
  "ד'",
  "thu";
  "ה'",
  "fri";
  "ו'",
  "sat";
  "ש'";
}
"wide";
{
  "sun";
  "יום ראשון",
  "mon";
  "יום שני",
```

```

        "tue";
        "יום שלישי",
        "wed";
        "יום רביעי",
        "thu";
        "יום חמישי",
        "fri";
        "יום שישי",
        "sat";
        "יום שבת";
    }
}
}
"quarters";
{
    "format";
    {
        "abbreviated";
        {
            "1";
            "Q1",
            "2";
            "Q2",
            "3";
            "Q3",
            "4";
            "Q4";
        }
        "narrow";
        {
            "1";
            "1",
            "2";
            "2",
            "3";
            "3",
            "4";
            "4";
        }
        "wide";
        {
            "1";
            "1 רבעון",
            "2";
            "2 רבעון",
            "3";
            "3 רבעון",
            "4";
            "4 רבעון";
        }
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
        "Q1",

```

```

        "2";
        "Q2",
        "3";
        "Q3",
        "4";
        "Q4";
    }
    "narrow";
    {
        "1";
        "1",
        "2";
        "2",
        "3";
        "3",
        "4";
        "4";
    }
    "wide";
    {
        "1";
        "1 רבעון",
        "2";
        "2 רבעון",
        "3";
        "3 רבעון",
        "4";
        "4 רבעון";
    }
}
}
"dayPeriods";
{
    "format";
    {
        "abbreviated";
        {
            "midnight";
            "חצות",
            "am";
            "לפנה"צ",
            "pm";
            "אחה"צ",
            "morning1";
            "בוקר",
            "afternoon1";
            "צהריים",
            "afternoon2";
            "אחר הצהריים",
            "evening1";
            "ערב",
            "night1";
            "לילה",
            "night2";
            "לפנות בוקר";
        }
    }
    "narrow";

```

```

    {
      "midnight";
      "חצות",
      "am";
      "לפנה"צ",
      "pm";
      "אחה"צ",
      "morning1";
      "בוקר",
      "afternoon1";
      "צהריים",
      "afternoon2";
      "אחר הצהריים",
      "evening1";
      "ערב",
      "night1";
      "לילה",
      "night2";
      "לפנות בוקר";
    }
    "wide";
    {
      "midnight";
      "חצות",
      "am";
      "לפנה"צ",
      "pm";
      "אחה"צ",
      "morning1";
      "בוקר",
      "afternoon1";
      "צהריים",
      "afternoon2";
      "אחר הצהריים",
      "evening1";
      "ערב",
      "night1";
      "לילה",
      "night2";
      "לפנות בוקר";
    }
  }
  "stand-alone";
  {
    "abbreviated";
    {
      "midnight";
      "חצות",
      "am";
      "לפנה"צ",
      "pm";
      "אחה"צ",
      "morning1";
      "בוקר",
      "afternoon1";
      "צהריים",
      "afternoon2";
    }
  }

```

```

        "אחר הצהריים",
        "evening1";
        "ערב",
        "night1";
        "לילה",
        "night2";
        "לפנות בוקר";
    }
    "narrow";
    {
        "midnight";
        "חצות",
        "am";
        "לפנה"צ",
        "pm";
        "אחה"צ",
        "morning1";
        "בוקר",
        "afternoon1";
        "צהריים",
        "afternoon2";
        "אחר הצהריים",
        "evening1";
        "ערב",
        "night1";
        "לילה",
        "night2";
        "לפנות בוקר";
    }
    "wide";
    {
        "midnight";
        "חצות",
        "am";
        "לפנה"צ",
        "pm";
        "אחה"צ",
        "morning1";
        "בוקר",
        "afternoon1";
        "צהריים",
        "afternoon2";
        "אחר הצהריים",
        "evening1";
        "ערב",
        "night1";
        "לילה",
        "night2";
        "לפנות בוקר";
    }
    }
    "eras";
    {
        "eraNames";
        {
            "0";

```

```

        "לפני הספירה",
        "0-alt-variant";
        "לפנה"ס",
        "1";
        "לספירה",
        "1-alt-variant";
        "CE";
    }
    "eraAbbr";
    {
        "0";
        "לפנה"ס",
        "0-alt-variant";
        "BCE",
        "1";
        "לספירה",
        "1-alt-variant";
        "CE";
    }
    "eraNarrow";
    {
        "0";
        "לפנה"ס",
        "0-alt-variant";
        "BCE",
        "1";
        "לספירה",
        "1-alt-variant";
        "CE";
    }
}
"dateFormats";
{
    "full";
    "EEEE, d MMMM y",
    "long";
    "d MMMM y",
    "medium";
    "d MMM y",
    "short";
    "d.M.y";
}
"timeFormats";
{
    "full";
    "H:mm:ss zzzz",
    "long";
    "H:mm:ss z",
    "medium";
    "H:mm:ss",
    "short";
    "H:mm";
}
"dateTimeFormats";
{
    "full";
    "{1} בשעה {0}",

```



```

        "long";
        "{1} בשעה {0}",
        "medium";
        "{1}, {0}",
        "short";
        "{1}, {0}",
        "availableFormats";
    {
        "d";
        "d",
        "E";
        "ccc",
        "Ed";
        "E ה-d",
        "Ehm";
        "E h:mm a",
        "EHm";
        "E H:mm",
        "Ehms";
        "E h:mm:ss a",
        "EHms";
        "E H:mm:ss",
        "Gy";
        "y G",
        "GyMMM";
        "MMM y G",
        "GyMMMd";
        "d הMMM y G",
        "GyMMMED";
        "E, d הMMM y G",
        "h";
        "h a",
        "H";
        "H",
        "hm";
        "h:mm a",
        "Hm";
        "H:mm",
        "hms";
        "h:mm:ss a",
        "Hms";
        "H:mm:ss",
        "hmsv";
        "h:mm:ss a v",
        "Hmsv";
        "HH:mm:ss v",
        "hmv";
        "h:mm a v",
        "Hmv";
        "HH:mm v",
        "M";
        "L",
        "Md";
        "d.M",
        "MED";
        "E, d.M",
        "MMM";
    }

```

```

    "LLL",
    "MMMd";
    "d בMMM",
    "MMMEd";
    "E, d בMMM",
    "MMMMd";
    "d בMMMM",
    "MMMMW-count-one";
    "שבוע W בMMM",
    "MMMMW-count-two";
    "שבוע W בMMM",
    "MMMMW-count-many";
    "שבוע W בMMM",
    "MMMMW-count-other";
    "שבוע W בMMM",
    "ms";
    "mm:ss",
    "y";
    "Y",
    "yM";
    "M.y",
    "yMd";
    "d.M.y",
    "yMEd";
    "E, d.M.y",
    "yMM";
    "M.y",
    "yMMM";
    "MMM y",
    "yMMMd";
    "d בMMM y",
    "yMMMEd";
    "E, d בMMM y",
    "yMMMM";
    "MMMM y",
    "yQQQ";
    "QQQ y",
    "yQQQQ";
    "QQQQ y",
    "yw-count-one";
    "שבוע w בשנת y",
    "yw-count-two";
    "שבוע w בשנת y",
    "yw-count-many";
    "שבוע w בשנת y",
    "yw-count-other";
    "שבוע w בשנת y";
  }
  "appendItems";
  {
    "Day";
    "{0} ({2}: {1})",
    "Day-Of-Week";
    "{0} {1}",
    "Era";
    "{1} {0}",
    "Hour";
  }

```

```

        "{0} ({2}: {1})",
        "Minute";
        "{0} ({2}: {1})",
        "Month";
        "{0} ({2}: {1})",
        "Quarter";
        "{0} ({2}: {1})",
        "Second";
        "{0} ({2}: {1})",
        "Timezone";
        "{0} {1}",
        "Week";
        "{0} ({2}: {1})",
        "Year";
        "{1} {0}";
    }
    "intervalFormats";
    {
        "intervalFormatFallback";
        "{0} - {1}",
        "d";
        {
            "d";
            "d-d";
        }
        "h";
        {
            "a";
            "h a - h a",
            "h";
            "h-h a";
        }
        "H";
        {
            "H";
            "H-H";
        }
        "hm";
        {
            "a";
            "h:mm a - h:mm a",
            "h";
            "h:mm-h:mm a",
            "m";
            "h:mm-h:mm a";
        }
        "Hm";
        {
            "H";
            "H:mm-H:mm",
            "m";
            "H:mm-H:mm";
        }
        "hmv";
        {
            "a";
            "h:mm a - h:mm a v",

```

```

        "h";
        "h:mm-h:mm a v",
        "m";
        "h:mm-h:mm a v";
    }
    "Hmv";
    {
        "H";
        "H:mm-H:mm v",
        "m";
        "H:mm-H:mm v";
    }
    "hv";
    {
        "a";
        "h a - h a v",
        "h";
        "h-h a v";
    }
    "Hv";
    {
        "H";
        "H-H v";
    }
    "M";
    {
        "M";
        "M-M";
    }
    "Md";
    {
        "d";
        "d.M-d.M",
        "M";
        "d.M-d.M";
    }
    "MED";
    {
        "d";
        "EEEE d.M-EEEE d.M",
        "M";
        "EEEE d.M - EEEE d.M";
    }
    "MMM";
    {
        "M";
        "MMM-MMM";
    }
    "MMMd";
    {
        "d";
        "d-d  MMM",
        "M";
        "d  MMM-d  MMM";
    }
    "MMMED";
    {

```

```

        "d";
        "EEEE, d 1MMM - EEEE, d 1MMM",
        "M";
        "EEEE, d 1MMM - EEEE, d 1MMM";
    }
    "MMMM";
    {
        "M";
        "LLLL-LLLL";
    }
    "Y";
    {
        "Y";
        "Y-Y";
    }
    "yM";
    {
        "M";
        "M.Y-M.Y",
        "Y";
        "M.Y-M.Y";
    }
    "yMd";
    {
        "d";
        "dd.M.Y - dd.M.Y",
        "M";
        "d.M.Y - d.M.Y",
        "Y";
        "d.M.Y - d.M.Y";
    }
    "yMEd";
    {
        "d";
        "EEEE d.M.Y - EEEE d.M.Y",
        "M";
        "EEEE d.M.Y - EEEE d.M.Y",
        "Y";
        "EEEE d.M.Y - EEEE d.M.Y";
    }
    "yMMM";
    {
        "M";
        "MMM-MMM Y",
        "Y";
        "MMM Y - MMM Y";
    }
    "yMMMd";
    {
        "d";
        "d-d 1MMM Y",
        "M";
        "d MMM - d MMM Y",
        "Y";
        "d MMM Y - d MMM Y";
    }
    "yMMMEd";

```

```

    }
    }
    }
    }
    }
    {
        "d";
        "EEEE d MMM - EEEE d MMM y",
        "M";
        "EEEE d MMM - EEEE d MMM y",
        "Y";
        "EEEE d MMM y - EEEE d MMM y";
    }
    "YMMMM";
    {
        "M";
        "MMMM-MMMM y",
        "Y";
        "MMMM y-MMMM Y";
    }
}

```

INDEX.JSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// Load the L10n, loadCldr from ej2-base
import { loadCldr, L10n } from '@syncfusion/ej2-base';
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
// load the CLDR data files.
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as timeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames);
L10n.load({
    'he': {
        'datepicker': { placeholder: 'הזן תאריך',
            today: 'היום'
        }
    }
});
// import the datepickercomponent
function App() {
    return <DatePickerComponent id="datepicker" locale='he'
enableRtl={true}/>;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
```

```
// Load the L10n, loadCldr from ej2-base
import { loadCldr, L10n } from '@syncfusion/ej2-base';
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
// load the CLDR data files.
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as timeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames);
L10n.load({
  'he': {
    'datepicker': { placeholder: 'הזן תאריך',
      today: 'היום'
    }
  }
});
// import the datepickercomponent
function App() {
  return <DatePickerComponent id="datepicker" locale='he' enableRtl={true}>
/>;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

NUMBERINGSYSTEMS.JSON

```
{
  "supplemental": {
    "version": {
      "_number": "$Revision: 12732 $",
      "_unicodeVersion": "9.0.0",
      "_cldrVersion": "31"
    },
    "numberingSystems": {
      "adlm": {
        "_digits": "ᲐᲑᲔᲕᲗᲘᲙᲚᲛᲞᲟᲡᲢᲣᲤᲥᲦᲧᲨᲩᲪᲫᲬᲭᲮᲯᲰᲱᲲᲳᲴᲵᲶᲷᲸᲹᲺ᲻᲼ᲽᲾᲿ",
        "_type": "numeric"
      },
      "ahom": {
        "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍᱎᱏ᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱱᱲᱴᱶᱰᱵᱽ᱾᱿",
        "_type": "numeric"
      },
      "arab": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "arabext": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "armn": {
        "_rules": "armenian-upper",
        "_type": "algorithmic"
      },
      "armnlow": {
        "_rules": "armenian-lower",

```

```
    "_type": "algorithmic"
  },
  "bali": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "beng": {
    "_digits": "০১২৩৪৫৬৭৮৯",
    "_type": "numeric"
  },
  "bhks": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "brah": {
    "_digits": "·\u094d\u0940\u094d\u0948\u094d\u094a\u094d\u094c\u094d\u094e\u094d\u094f",
    "_type": "numeric"
  },
  "cakm": {
    "_digits": "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
    "_type": "numeric"
  },
  "cham": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "cyr1": {
    "_rules": "cyrillic-lower",
    "_type": "algorithmic"
  },
  "deva": {
    "_digits": "०१२३४५६७८९",
    "_type": "numeric"
  },
  "ethi": {
    "_rules": "ethiopic",
    "_type": "algorithmic"
  },
  "fullwide": {
    "_digits": "0 1 2 3 4 5 6 7 8 9",
    "_type": "numeric"
  },
  "geor": {
    "_rules": "georgian",
    "_type": "algorithmic"
  },
  "grek": {
    "_rules": "greek-upper",
    "_type": "algorithmic"
  },
  "greklow": {
    "_rules": "greek-lower",
    "_type": "algorithmic"
  },
  "gujr": {
```



```

    "_digits": "၀၇၃၃၄၆၇၉",
    "_type": "numeric"
  },
  "guru": {
    "_digits": "၀၇၃၃၄၆၇၉",
    "_type": "numeric"
  },
  "hanidays": {
    "_rules": "zh/SpelloutRules/spellout-numbering-days",
    "_type": "algorithmic"
  },
  "hanidec": {
    "_digits": "〇一二三四五六七八九",
    "_type": "numeric"
  },
  "hans": {
    "_rules": "zh/SpelloutRules/spellout-cardinal",
    "_type": "algorithmic"
  },
  "hansfin": {
    "_rules": "zh/SpelloutRules/spellout-cardinal-financial",
    "_type": "algorithmic"
  },
  "hant": {
    "_rules": "zh_Hant/SpelloutRules/spellout-cardinal",
    "_type": "algorithmic"
  },
  "hantfin": {
    "_rules": "zh_Hant/SpelloutRules/spellout-cardinal-financial",
    "_type": "algorithmic"
  },
  "hebr": {
    "_rules": "hebrew",
    "_type": "algorithmic"
  },
  "hmng": {
    "_digits": "᠐ᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨ",
    "_type": "numeric"
  },
  "java": {
    "_digits": "᠐ᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨ",
    "_type": "numeric"
  },
  "jpan": {
    "_rules": "ja/SpelloutRules/spellout-cardinal",
    "_type": "algorithmic"
  },
  "jpanfin": {
    "_rules": "ja/SpelloutRules/spellout-cardinal-financial",
    "_type": "algorithmic"
  },
  "kali": {
    "_digits": "᠐ᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨ",
    "_type": "numeric"
  },

```

```
"khmr": {
  "_digits": "០១២៣៤៥៦៧៨៩",
  "_type": "numeric"
},
"knda": {
  "_digits": "೦೧೨೩೪೫೬೭೮೯",
  "_type": "numeric"
},
"lana": {
  "_digits": "□□□□□□□□",
  "_type": "numeric"
},
"lanatham": {
  "_digits": "□□□□□□□□",
  "_type": "numeric"
},
"laoo": {
  "_digits": "໐໑໒໓໔໕໖໗໘໑",
  "_type": "numeric"
},
"latn": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"lepc": {
  "_digits": "□□□□□□□□",
  "_type": "numeric"
},
"limb": {
  "_digits": "□□□□□□□□",
  "_type": "numeric"
},
"mathbold": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathdbl": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathmono": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathsanb": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathsans": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mlym": {
  "_digits": "൦൧൨൩൪൫൬൭൮൯",
  "_type": "numeric"
}
```

```

    },
    "modi": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    "mong": {
      "_digits": "᠐᠑᠒᠓᠔᠕᠖᠗᠘᠐",
      "_type": "numeric"
    },
    "mroo": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    "mtei": {
      "_digits": "᠐᠑ᠶ᠋ᠩᠸ᠋ᠱ᠋ᠰ᠋ᠴ᠋ᠠᠨ",
      "_type": "numeric"
    },
    "mymr": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "mymrshan": {
      "_digits": "0123456789",
      "_type": "numeric"
    },
    "mymrtlng": {
      "_digits": "᠐ᠠᠨᠢᠵᠤᠨᠢᠯᠠᠩᠭᠤᠨ",
      "_type": "numeric"
    },
    "newa": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    "nkoo": {
      "_digits": "᠐ᠠᠨᠠᠭᠤᠨᠠᠭᠤᠨ",
      "_type": "numeric"
    },
    "olck": {
      "_digits": "᠐ᠠᠨᠠᠭᠤᠨᠠᠭᠤᠨ",
      "_type": "numeric"
    },
    "orya": {
      "_digits": "୦୧୨୩୪୫୬୭୮୯",
      "_type": "numeric"
    },
    "osma": {
      "_digits": "᠐᠑ᠶ᠋ᠩᠸ᠋ᠱ᠋ᠰ᠋ᠴ᠋ᠠᠨ",
      "_type": "numeric"
    },
    "roman": {
      "_rules": "roman-upper",
      "_type": "algorithmic"
    },
  },

```

```

"romanlow": {
  "_rules": "roman-lower",
  "_type": "algorithmic"
},
"saur": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"shrd": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"sind": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"sinh": {
  "_digits": "⁂௧௧௧௧௧௧௧௧௧௧௧",
  "_type": "numeric"
},
"sora": {
  "_digits": "0௧23456789",
  "_type": "numeric"
},
"sund": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"takr": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"talv": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"taml": {
  "_rules": "tamil",
  "_type": "algorithmic"
},
"tamldec": {
  "_digits": "0௧௨௩௪௫௬௭௮௯",
  "_type": "numeric"
},
"telu": {
  "_digits": "0౧౨౩౪౫౬౭౮౯",
  "_type": "numeric"
},
"thai": {
  "_digits": "๐๑๒๓๔๕๖๗๘๙",
  "_type": "numeric"
},
"tibv": {
  "_digits": "0123456789",
  "_type": "numeric"
}

```

```
{  
    "tirh": {  
        "_digits": "□□□□□□□□□□",  
        "_type": "numeric"  
    },  
    "vairi": {  
        "_digits": "ྐ|འ་རྩིས་དོན་ཅུག་",  
        "_type": "numeric"  
    },  
    "wara": {  
        "_digits": "□□□□□□□□□□",  
        "_type": "numeric"  
    }  
}
```

NUMBERINGSYSTEMS.JSX

```
{
  "supplemental";
  {
    "version";
    {
      "_number";
      "$Revision: 12732 $",
      "_unicodeVersion";
      "9.0.0",
      "_cldrVersion";
      "31";
    }
    "numberingSystems";
    {
      "adlm";
      {
        "_digits";
        "୧୨୩୪୫୬୭୮୯୦",
        "_type";
        "numeric";
      }
      "ahom";
      {
        "_digits";
        "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍᱎᱏ᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱱᱲᱳᱴᱵᱶᱷᱸᱹᱺᱻᱼᱽ᱾᱿",
        "_type";
        "numeric";
      }
      "arab";
      {
        "_digits";
        "٠١٢٣٤٥٦٧٨٩",
        "_type";
        "numeric";
      }
      "arabext";
      {
```

```

        "_digits";
        "٠١٢٣٤٥٦٧٨٩",
        "_type";
        "numeric";
    }
    "armn";
    {
        "_rules";
        "armenian-upper",
        "_type";
        "algorithmic";
    }
    "armnlow";
    {
        "_rules";
        "armenian-lower",
        "_type";
        "algorithmic";
    }
    "bali";
    {
        "_digits";
        "ᮀᮁᮂᮃᮄᮅᮆᮇᮈᮉ",
        "_type";
        "numeric";
    }
    "beng";
    {
        "_digits";
        "০১২৩৪৫৬৭৮৯",
        "_type";
        "numeric";
    }
    "bhks";
    {
        "_digits";
        "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉",
        "_type";
        "numeric";
    }
    "brah";
    {
        "_digits";
        "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉",
        "_type";
        "numeric";
    }
    "cakm";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "cham";
    {

```

```

        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "cyr1";
    {
        "_rules";
        "cyrillic-lower",
        "_type";
        "algorithmic";
    }
    "deva";
    {
        "_digits";
        "ॐ१२३४५६७८९",
        "_type";
        "numeric";
    }
    "ethi";
    {
        "_rules";
        "ethiopic",
        "_type";
        "algorithmic";
    }
    "fullwide";
    {
        "_digits";
        "0 1 2 3 4 5 6 7 8 9",
        "_type";
        "numeric";
    }
    "geor";
    {
        "_rules";
        "georgian",
        "_type";
        "algorithmic";
    }
    "grek";
    {
        "_rules";
        "greek-upper",
        "_type";
        "algorithmic";
    }
    "greklow";
    {
        "_rules";
        "greek-lower",
        "_type";
        "algorithmic";
    }
    "gujr";
    {

```

```

        "_digits";
        "୦୧୨୩୪୫୬୭୮୯",
        "_type";
        "numeric";
    }
    "guru";
    {
        "_digits";
        "୦୧୨୩୪୫୬୭୮୯",
        "_type";
        "numeric";
    }
    "hanidays";
    {
        "_rules";
        "zh/SpelloutRules/spellout-numbering-days",
        "_type";
        "algorithmic";
    }
    "hanidec";
    {
        "_digits";
        "〇一二三四五六七八九",
        "_type";
        "numeric";
    }
    "hans";
    {
        "_rules";
        "zh/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "hansfin";
    {
        "_rules";
        "zh/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "hant";
    {
        "_rules";
        "zh_Hant/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "hantfin";
    {
        "_rules";
        "zh_Hant/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "hebr";
    {

```



```

        "_rules";
        "hebrew",
        "_type";
        "algorithmic";
    }
    "hmng";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
    "java";
    {
        "_digits";
        "௦௧௨௩௪௫௬௭௮௯௦௧௨௩௪௫௬௭௮௯௦",
        "_type";
        "numeric";
    }
    "jpan";
    {
        "_rules";
        "ja/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "jpanfin";
    {
        "_rules";
        "ja/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "kali";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
    "khmr";
    {
        "_digits";
        "០១២៣៤៥៦៧៨៩",
        "_type";
        "numeric";
    }
    "knda";
    {
        "_digits";
        "೦೧೨೩೪೫೬೭೮೯",
        "_type";
        "numeric";
    }
    "lana";

```

```

    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "lanatham";
    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "laoo";
    {
      "_digits";
      "໐໑໒໓໔໕໖໗໘໑",
      "_type";
      "numeric";
    }
    "latn";
    {
      "_digits";
      "0123456789",
      "_type";
      "numeric";
    }
    "lepc";
    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "limb";
    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "mathbold";
    {
      "_digits";
      "0123456789",
      "_type";
      "numeric";
    }
    "mathdbl";
    {
      "_digits";
      "0123456789",
      "_type";
      "numeric";
    }
    "mathmono";

```

```

    {
      "_digits";
      "0123456789",
      "_type";
      "numeric";
    }
    "mathsanb";
    {
      "_digits";
      "0123456789",
      "_type";
      "numeric";
    }
    "mathsans";
    {
      "_digits";
      "0123456789",
      "_type";
      "numeric";
    }
    "mlym";
    {
      "_digits";
      "ഐഹനരൂനവുൻ",
      "_type";
      "numeric";
    }
    "modi";
    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "mong";
    {
      "_digits";
      "᠐᠑᠒᠓᠐ᠠᠨᠨᠠᠭ",
      "_type";
      "numeric";
    }
    "mroo";
    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "mtei";
    {
      "_digits";
      "၆၆၆၆၆၆၆၆၆",
      "_type";
      "numeric";
    }
    "mymr";

```

```

    {
      "_digits";
      "၀၁၂၃၄၅၆၇၈",
      "_type";
      "numeric";
    }
    "mymrshan";
    {
      "_digits";
      "၀၁၂၃၄၅၆၇၈",
      "_type";
      "numeric";
    }
    "mymrtlng";
    {
      "_digits";
      "၀၁၂၃၄၅၆၇၈",
      "_type";
      "numeric";
    }
    "newa";
    {
      "_digits";
      "၀၁၂၃၄၅၆၇၈",
      "_type";
      "numeric";
    }
    "nkoo";
    {
      "_digits";
      "၀၁၂၃၄၅၆၇၈",
      "_type";
      "numeric";
    }
    "olck";
    {
      "_digits";
      "၀၁၂၃၄၅၆၇၈",
      "_type";
      "numeric";
    }
    "orya";
    {
      "_digits";
      "၀၁၂၃၄၅၆၇၈",
      "_type";
      "numeric";
    }
    "osma";
    {
      "_digits";
      "၀၁၂၃၄၅၆၇၈",
      "_type";
      "numeric";
    }
  }

```

```

    }
    "roman";
    {
        "_rules";
        "roman-upper",
        "_type";
        "algorithmic";
    }
    "romanlow";
    {
        "_rules";
        "roman-lower",
        "_type";
        "algorithmic";
    }
    "saur";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
    "shrd";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
    "sind";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
    "sinh";
    {
        "_digits";
        "𑆑𑆒𑆓𑆔𑆕𑆖𑆗𑆘𑆙𑆚",
        "_type";
        "numeric";
    }
    "sora";
    {
        "_digits";
        "᱆ᱏᱟᱨᱢᱟᱝ",
        "_type";
        "numeric";
    }
    "sund";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }

```

```

    }
    "takr";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
    "talu";
    {
        "_digits";
        "ᱠᱟᱱᱟᱢᱟᱞᱟᱢ",
        "_type";
        "numeric";
    }
    "taml";
    {
        "_rules";
        "tamil",
        "_type";
        "algorithmic";
    }
    "tamldec";
    {
        "_digits";
        "௦௧௨௩௪௫௬௭௮௯",
        "_type";
        "numeric";
    }
    "telu";
    {
        "_digits";
        "౦౧౨౩౪౫౬౭౮౯",
        "_type";
        "numeric";
    }
    "thai";
    {
        "_digits";
        "๐๑๒๓๔๕๖๗๘๙",
        "_type";
        "numeric";
    }
    "tib";
    {
        "_digits";
        "༠༡༢༣༤༥༦༧༨༩",
        "_type";
        "numeric";
    }
    "tirh";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }

```

```

    }
    "vairi";
    {
        "_digits";
        "᠑᠒᠓᠔᠕᠖᠗᠘᠐᠐᠑᠑",
        "_type";
        "numeric";
    }
    "wara";
    {
        "_digits";
        "᠑᠒᠓᠔᠕᠖᠗᠘᠐᠐᠑᠑",
        "_type";
        "numeric";
    }
    }
}

```

NUMBERS.JSON

```

{
  "main": {
    "he": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31.0.1"
        },
        "language": "he"
      },
      "numbers": {
        "defaultNumberingSystem": "latn",
        "otherNumberingSystems": {
          "native": "latn",
          "traditional": "hebr"
        },
        "minimumGroupingDigits": "1",
        "symbols-numberSystem-latn": {
          "decimal": ".",
          "group": ",",
          "list": ";",
          "percentSign": "%",
          "plusSign": "+",
          "minusSign": "-",
          "exponential": "E",
          "superscriptingExponent": "x",
          "perMille": "‰",
          "infinity": "∞",
          "nan": "NaN",
          "timeSeparator": ":"
        },
        "decimalFormats-numberSystem-latn": {
          "standard": "#,##0.###",
          "long": {
            "decimalFormat": {

```

```

    "1000-count-one": "אלף 0",
    "1000-count-two": "אלף 0",
    "1000-count-many": "אלף 0",
    "1000-count-other": "אלף 0",
    "10000-count-one": "אלף 00",
    "10000-count-two": "אלף 00",
    "10000-count-many": "אלף 00",
    "10000-count-other": "אלף 00",
    "100000-count-one": "אלף 000",
    "100000-count-two": "אלף 000",
    "100000-count-many": "אלף 000",
    "100000-count-other": "אלף 000",
    "1000000-count-one": "0 מיליון",
    "1000000-count-two": "0 מיליון",
    "1000000-count-many": "0 מיליון",
    "1000000-count-other": "0 מיליון",
    "10000000-count-one": "00 מיליון",
    "10000000-count-two": "00 מיליון",
    "10000000-count-many": "00 מיליון",
    "10000000-count-other": "00 מיליון",
    "100000000-count-one": "000 מיליון",
    "100000000-count-two": "000 מיליון",
    "100000000-count-many": "000 מיליון",
    "100000000-count-other": "000 מיליון",
    "1000000000-count-one": "0 מיליארד",
    "1000000000-count-two": "0 מיליארד",
    "1000000000-count-many": "0 מיליארד",
    "1000000000-count-other": "0 מיליארד",
    "10000000000-count-one": "00 מיליארד",
    "10000000000-count-two": "00 מיליארד",
    "10000000000-count-many": "00 מיליארד",
    "10000000000-count-other": "00 מיליארד",
    "100000000000-count-one": "000 מיליארד",
    "100000000000-count-two": "000 מיליארד",
    "100000000000-count-many": "000 מיליארד",
    "100000000000-count-other": "000 מיליארד",
    "1000000000000-count-one": "0 טריליון",
    "1000000000000-count-two": "0 טריליון",
    "1000000000000-count-many": "0 טריליון",
    "1000000000000-count-other": "0 טריליון",
    "10000000000000-count-one": "00 טריליון",
    "10000000000000-count-two": "00 טריליון",
    "10000000000000-count-many": "00 טריליון",
    "10000000000000-count-other": "00 טריליון",
    "100000000000000-count-one": "000 טריליון",
    "100000000000000-count-two": "000 טריליון",
    "100000000000000-count-many": "000 טריליון",
    "100000000000000-count-other": "000 טריליון"
  },
  },
  "short": {
    "decimalFormat": {
      "1000-count-one": "0K",
      "1000-count-two": "0K",
      "1000-count-many": "0K",
      "1000-count-other": "0K",
      "10000-count-one": "00K",
    }
  }

```



```

        "10000-count-two": "00K",
        "10000-count-many": "00K",
        "10000-count-other": "00K",
        "100000-count-one": "000K",
        "100000-count-two": "000K",
        "100000-count-many": "000K",
        "100000-count-other": "000K",
        "1000000-count-one": "0M",
        "1000000-count-two": "0M",
        "1000000-count-many": "0M",
        "1000000-count-other": "0M",
        "10000000-count-one": "00M",
        "10000000-count-two": "00M",
        "10000000-count-many": "00M",
        "10000000-count-other": "00M",
        "100000000-count-one": "000M",
        "100000000-count-two": "000M",
        "100000000-count-many": "000M",
        "100000000-count-other": "000M",
        "1000000000-count-one": "0B",
        "1000000000-count-two": "0B",
        "1000000000-count-many": "0B",
        "1000000000-count-other": "0B",
        "10000000000-count-one": "00B",
        "10000000000-count-two": "00B",
        "10000000000-count-many": "00B",
        "10000000000-count-other": "00B",
        "100000000000-count-one": "000B",
        "100000000000-count-two": "000B",
        "100000000000-count-many": "000B",
        "100000000000-count-other": "000B",
        "1000000000000-count-one": "0T",
        "1000000000000-count-two": "0T",
        "1000000000000-count-many": "0T",
        "1000000000000-count-other": "0T",
        "10000000000000-count-one": "00T",
        "10000000000000-count-two": "00T",
        "10000000000000-count-many": "00T",
        "10000000000000-count-other": "00T",
        "100000000000000-count-one": "000T",
        "100000000000000-count-two": "000T",
        "100000000000000-count-many": "000T",
        "100000000000000-count-other": "000T"
    }
}
},
"scientificFormats-numberSystem-latn": {
    "standard": "#E0"
},
"percentFormats-numberSystem-latn": {
    "standard": "#,##0%"
},
"currencyFormats-numberSystem-latn": {
    "currencySpacing": {
        "beforeCurrency": {
            "currencyMatch": "[:^S:]",
            "surroundingMatch": "[:digit:]",

```

```

        "insertBetween": " "
    },
    "afterCurrency": {
        "currencyMatch": "[:^S:]",
        "surroundingMatch": "[:digit:]",
        "insertBetween": " "
    }
},
"standard": "##0.00#, -¤; ##0.00, # ¤",
"accounting": "#, ##0.00 ¤",
"short": {
    "standard": {
        "1000-count-one": "¤ 0K",
        "1000-count-two": "¤ 0K",
        "1000-count-many": "¤0K",
        "1000-count-other": "¤ 0K",
        "10000-count-one": "¤00K",
        "10000-count-two": "¤00K",
        "10000-count-many": "¤00K",
        "10000-count-other": "¤ 00K",
        "100000-count-one": "¤000K",
        "100000-count-two": "¤000K",
        "100000-count-many": "¤000K",
        "100000-count-other": "¤000K",
        "1000000-count-one": "¤0M",
        "1000000-count-two": "¤0M",
        "1000000-count-many": "¤0M",
        "1000000-count-other": "¤0M",
        "10000000-count-one": "¤00M",
        "10000000-count-two": "¤00M",
        "10000000-count-many": "¤00M",
        "10000000-count-other": "¤00M",
        "100000000-count-one": "¤000M",
        "100000000-count-two": "¤000M",
        "100000000-count-many": "¤000M",
        "100000000-count-other": "¤000M",
        "1000000000-count-one": "¤0B",
        "1000000000-count-two": "¤0B",
        "1000000000-count-many": "¤0B",
        "1000000000-count-other": "¤0B",
        "10000000000-count-one": "¤00B",
        "10000000000-count-two": "¤00B",
        "10000000000-count-many": "¤00B",
        "10000000000-count-other": "¤00B",
        "100000000000-count-one": "¤000B",
        "100000000000-count-two": "¤000B",
        "100000000000-count-many": "¤000B",
        "100000000000-count-other": "¤000B",
        "1000000000000-count-one": "¤0T",
        "1000000000000-count-two": "¤0T",
        "1000000000000-count-many": "¤0T",
        "1000000000000-count-other": "¤0T",
        "10000000000000-count-one": "¤00T",
        "10000000000000-count-two": "¤00T",
        "10000000000000-count-many": "¤00T",
        "10000000000000-count-other": "¤00T",
        "100000000000000-count-one": "¤000T",
    }
}

```

```

        "1000000000000000-count-two": "א000T",
        "1000000000000000-count-many": "א000T",
        "1000000000000000-count-other": "א000T"
    },
    },
    "unitPattern-count-one": "{0} {1}",
    "unitPattern-count-two": "{0} {1}",
    "unitPattern-count-many": "{0} {1}",
    "unitPattern-count-other": "{0} {1}"
},
"miscPatterns-numberSystem-latn": {
    "atLeast": "≥{0}+",
    "range": "{0}-{1}"
},
"minimalPairs": {
    "pluralMinimalPairs": "שנה",
    "pluralMinimalPairs": "שנתיים",
    "pluralMinimalPairs": "{0} שנה",
    "pluralMinimalPairs": "{0} שנים",
    "other": "פנה ימיונה בפנייה ה-{0}"
}
}
}
}
}
}

```

NUMBERS.JSX

```

{
    "main";
    {
        "he";
        {
            "identity";
            {
                "version";
                {
                    "_number";
                    "$Revision: 13259 $",
                    "_cldrVersion";
                    "31.0.1";
                }
                "language";
                "he";
            }
            "numbers";
            {
                "defaultNumberingSystem";
                "latn",
                "otherNumberingSystems";
                {
                    "native";
                    "latn",
                    "traditional";
                    "hebr";
                }
            }
        }
    }
}

```

```

    "minimumGroupingDigits";
    "1",
    "symbols-numberSystem-latn";
    {
        "decimal";
        ".",
        "group";
        ",",
        "list";
        ";",
        "percentSign";
        "%",
        "plusSign";
        "+",
        "minusSign";
        "-",
        "exponential";
        "E",
        "superscriptingExponent";
        "×",
        "perMille";
        "‰",
        "infinity";
        "∞",
        "nan";
        "NaN",
        "timeSeparator";
        ":";
    }
    "decimalFormats-numberSystem-latn";
    {
        "standard";
        "#,##0.###",
        "long";
        {
            "decimalFormat";
            {
                "1000-count-one";
                "q7x 0",
                "1000-count-two";
                "q7x 0",
                "1000-count-many";
                "q7x 0",
                "1000-count-other";
                "q7x 0",
                "10000-count-one";
                "q7x 00",
                "10000-count-two";
                "q7x 00",
                "10000-count-many";
                "q7x 00",
                "10000-count-other";
                "q7x 00",
                "100000-count-one";
                "q7x 000",
                "100000-count-two";
                "q7x 000",
            }
        }
    }

```

```
"100000-count-many";
"אלף 000",
"100000-count-other";
"אלף 000",
"1000000-count-one";
"מיליון 0",
"1000000-count-two";
"מיליון 0",
"1000000-count-many";
"מיליון 0",
"1000000-count-other";
"מיליון 0",
"10000000-count-one";
"מיליון 00",
"10000000-count-two";
"מיליון 00",
"10000000-count-many";
"מיליון 00",
"10000000-count-other";
"מיליון 00",
"100000000-count-one";
"מיליון 000",
"100000000-count-two";
"מיליון 000",
"100000000-count-many";
"מיליון 000",
"100000000-count-other";
"מיליון 000",
"1000000000-count-one";
"מיליארד 0",
"1000000000-count-two";
"מיליארד 0",
"1000000000-count-many";
"מיליארד 0",
"1000000000-count-other";
"מיליארד 0",
"10000000000-count-one";
"מיליארד 00",
"10000000000-count-two";
"מיליארד 00",
"10000000000-count-many";
"מיליארד 00",
"10000000000-count-other";
"מיליארד 00",
"10000000000-count-one";
"מיליארד 000",
"10000000000-count-two";
"מיליארד 000",
"10000000000-count-many";
"מיליארד 000",
"10000000000-count-other";
"מיליארד 000",
"100000000000-count-one";
"טריליון 0",
"100000000000-count-two";
"טריליון 0",
"100000000000-count-many";
```

```

        "טריליון 0",
        "1000000000000-count-other";
        "טריליון 0",
        "1000000000000-count-one";
        "טריליון 00",
        "1000000000000-count-two";
        "טריליון 00",
        "1000000000000-count-many";
        "טריליון 00",
        "1000000000000-count-other";
        "טריליון 00",
        "1000000000000-count-one";
        "טריליון 000",
        "1000000000000-count-two";
        "טריליון 000",
        "1000000000000-count-many";
        "טריליון 000",
        "1000000000000-count-other";
        "טריליון 000";
    }
}
"short";
{
    "decimalFormat";
    {
        "1000-count-one";
        "0K",
        "1000-count-two";
        "0K",
        "1000-count-many";
        "0K",
        "1000-count-other";
        "0K",
        "10000-count-one";
        "00K",
        "10000-count-two";
        "00K",
        "10000-count-many";
        "00K",
        "10000-count-other";
        "00K",
        "100000-count-one";
        "000K",
        "100000-count-two";
        "000K",
        "100000-count-many";
        "000K",
        "100000-count-other";
        "000K",
        "1000000-count-one";
        "0M",
        "1000000-count-two";
        "0M",
        "1000000-count-many";
        "0M",
        "1000000-count-other";
        "0M",
    }
}

```

```
        "10000000-count-one";
    "00M",
        "10000000-count-two";
    "00M",
        "10000000-count-many";
    "00M",
        "10000000-count-other";
    "00M",
        "100000000-count-one";
    "000M",
        "100000000-count-two";
    "000M",
        "100000000-count-many";
    "000M",
        "100000000-count-other";
    "000M",
        "1000000000-count-one";
    "0B",
        "1000000000-count-two";
    "0B",
        "1000000000-count-many";
    "0B",
        "1000000000-count-other";
    "0B",
        "10000000000-count-one";
    "00B",
        "10000000000-count-two";
    "00B",
        "10000000000-count-many";
    "00B",
        "10000000000-count-other";
    "00B",
        "100000000000-count-one";
    "000B",
        "100000000000-count-two";
    "000B",
        "100000000000-count-many";
    "000B",
        "100000000000-count-other";
    "000B",
        "1000000000000-count-one";
    "0T",
        "1000000000000-count-two";
    "0T",
        "1000000000000-count-many";
    "0T",
        "1000000000000-count-other";
    "0T",
        "10000000000000-count-one";
    "00T",
        "10000000000000-count-two";
    "00T",
        "10000000000000-count-many";
    "00T",
        "10000000000000-count-other";
    "00T",
        "100000000000000-count-one";
    "000T",
        "100000000000000-count-two";
    "000T",
        "100000000000000-count-many";
    "000T",
        "100000000000000-count-other";
    "000T",
        "1000000000000000-count-one";
```

```

        "000T",
        "1000000000000000-count-two";
        "000T",
        "1000000000000000-count-many";
        "000T",
        "1000000000000000-count-other";
        "000T";
    }
}
}
"scientificFormats-numberSystem-latn";
{
    "standard";
    "#E0";
}
"percentFormats-numberSystem-latn";
{
    "standard";
    "#,##0%";
}
"currencyFormats-numberSystem-latn";
{
    "currencySpacing";
    {
        "beforeCurrency";
        {
            "currencyMatch";
            "[:^S:]",
            "surroundingMatch";
            "[:digit:]",
            "insertBetween";
            " ";
        }
        "afterCurrency";
        {
            "currencyMatch";
            "[:^S:]",
            "surroundingMatch";
            "[:digit:]",
            "insertBetween";
            " ";
        }
    }
}
"standard";
"##0.00#, -¤; ##0.00#, ¤",
    "accounting";
"#,##0.00 ¤",
    "short";
{
    "standard";
    {
        "1000-count-one";
        "¤ 0K",
        "1000-count-two";
        "¤ 0K",
        "1000-count-many";
        "¤0K",
    }
}

```



```
        "1000-count-other";
    "¤ 0K",
        "10000-count-one";
    "¤00K",
        "10000-count-two";
    "¤00K",
        "10000-count-many";
    "¤00K",
        "10000-count-other";
    "¤ 00K",
        "100000-count-one";
    "¤000K",
        "100000-count-two";
    "¤000K",
        "100000-count-many";
    "¤000K",
        "100000-count-other";
    "¤000K",
        "1000000-count-one";
    "¤0M",
        "1000000-count-two";
    "¤0M",
        "1000000-count-many";
    "¤0M",
        "1000000-count-other";
    "¤0M",
        "10000000-count-one";
    "¤00M",
        "10000000-count-two";
    "¤00M",
        "10000000-count-many";
    "¤00M",
        "10000000-count-other";
    "¤00M",
        "100000000-count-one";
    "¤000M",
        "100000000-count-two";
    "¤000M",
        "100000000-count-many";
    "¤000M",
        "100000000-count-other";
    "¤000M",
        "1000000000-count-one";
    "¤0B",
        "1000000000-count-two";
    "¤0B",
        "1000000000-count-many";
    "¤0B",
        "1000000000-count-other";
    "¤0B",
        "10000000000-count-one";
    "¤00B",
        "10000000000-count-two";
    "¤00B",
        "10000000000-count-many";
    "¤00B",
        "10000000000-count-other";
```

```

        "א00B",
        "100000000000-count-one";
        "א000B",
        "100000000000-count-two";
        "א000B",
        "100000000000-count-many";
        "א000B",
        "100000000000-count-other";
        "א000B",
        "100000000000-count-one";
        "א0T",
        "100000000000-count-two";
        "א0T",
        "100000000000-count-many";
        "א0T",
        "100000000000-count-other";
        "א0T",
        "100000000000-count-one";
        "א00T",
        "100000000000-count-two";
        "א00T",
        "100000000000-count-many";
        "א00T",
        "100000000000-count-other";
        "א00T",
        "100000000000-count-one";
        "א000T",
        "100000000000-count-two";
        "א000T",
        "100000000000-count-many";
        "א000T",
        "100000000000-count-other";
        "א000T";
    }
}
"unitPattern-count-one";
"{0} {1}",
    "unitPattern-count-two";
"{0} {1}",
    "unitPattern-count-many";
"{0} {1}",
    "unitPattern-count-other";
"{0} {1}";
}
"miscPatterns-numberSystem-latn";
{
    "atLeast";
    "≥{0}+",
        "range";
    "{0}-{1}";
}
"minimalPairs";
{
    "pluralMinimalPairs";
    "שנה",
        "pluralMinimalPairs";
    "שנתיים",

```

```

        "pluralMinimalPairs";
        "{0} שנה",
        "pluralMinimalPairs";
        "{0} שנים",
        "other";
        "פנה ימינה בפנייה ה-{0}";
    }
}
}
}
}

```

TIMEZONENAMES.JSON

```

{
  "main": {
    "he": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31.0.1"
        },
        "language": "he"
      },
      "dates": {
        "timeZoneNames": {
          "hourFormat": "+HH:mm;-HH:mm",
          "gmtFormat": "GMT{0}",
          "gmtZeroFormat": "GMT",
          "regionFormat": "שעון {0}",
          "regionFormat-type-daylight": "קיצ {0} (שעון",
          "regionFormat-type-standard": "חורף {0} (שעון",
          "fallbackFormat": "{1} ({0})",
          "zone": {
            "America": {
              "Adak": {
                "exemplarCity": "אדאק"
              },
              "Anchorage": {
                "exemplarCity": "אנקורג'י"
              },
              "Anguilla": {
                "exemplarCity": "אנגווילה"
              },
              "Antigua": {
                "exemplarCity": "אנטיגואה"
              },
              "Araguaina": {
                "exemplarCity": "אראגואינה"
              },
              "Argentina": {
                "Rio_Gallegos": {
                  "exemplarCity": "ריו גאייגוס"
                },
                "San_Juan": {
                  "exemplarCity": "סן חואן"
                }
              }
            }
          }
        }
      }
    }
  }
}

```

```
    },  
    "Ushuaia": {  
      "exemplarCity": "אושואיה"  
    },  
    "La_Rioja": {  
      "exemplarCity": "לה ריוחה"  
    },  
    "San_Luis": {  
      "exemplarCity": "סן לואיס"  
    },  
    "Salta": {  
      "exemplarCity": "סלטה"  
    },  
    "Tucuman": {  
      "exemplarCity": "טוקומן"  
    }  
  },  
  "Aruba": {  
    "exemplarCity": "ארובה"  
  },  
  "Asuncion": {  
    "exemplarCity": "אסונסיון"  
  },  
  "Bahia": {  
    "exemplarCity": "באהיה"  
  },  
  "Bahia_Banderas": {  
    "exemplarCity": "באהיה בנדרס"  
  },  
  "Barbados": {  
    "exemplarCity": "ברבדוס"  
  },  
  "Belem": {  
    "exemplarCity": "בלם"  
  },  
  "Belize": {  
    "exemplarCity": "בליז"  
  },  
  "Blanc-Sablon": {  
    "exemplarCity": "בלאן-סבלון"  
  },  
  "Boa_Vista": {  
    "exemplarCity": "בואה ויסטה"  
  },  
  "Bogota": {  
    "exemplarCity": "בוגוטה"  
  },  
  "Boise": {  
    "exemplarCity": "בויסי"  
  },  
  "Buenos_Aires": {  
    "exemplarCity": "בואנוס איירס"  
  },  
  "Cambridge_Bay": {  
    "exemplarCity": "קיימברידג' ביי"  
  },  
  "Campo_Grande": {
```

```
    "exemplarCity": "קמפּו גרנדה"  
  },  
  "Cancun": {  
    "exemplarCity": "קנקון"  
  },  
  "Caracas": {  
    "exemplarCity": "קראקס"  
  },  
  "Catamarca": {  
    "exemplarCity": "קטמרקה"  
  },  
  "Cayenne": {  
    "exemplarCity": "קאייין"  
  },  
  "Cayman": {  
    "exemplarCity": "קיימן"  
  },  
  "Chicago": {  
    "exemplarCity": "שיקגו"  
  },  
  "Chihuahua": {  
    "exemplarCity": "צ'יוואווה"  
  },  
  "Coral_Harbour": {  
    "exemplarCity": "אטיקוקן"  
  },  
  "Cordoba": {  
    "exemplarCity": "קורדובה"  
  },  
  "Costa_Rica": {  
    "exemplarCity": "קוסטה ריקה"  
  },  
  "Creston": {  
    "exemplarCity": "קרטסון"  
  },  
  "Cuiaba": {  
    "exemplarCity": "קויאבה"  
  },  
  "Curacao": {  
    "exemplarCity": "קוראסאו"  
  },  
  "Danmarkshavn": {  
    "exemplarCity": "דנמרקסהוון"  
  },  
  "Dawson": {  
    "exemplarCity": "דוסון"  
  },  
  "Dawson_Creek": {  
    "exemplarCity": "דוסון קריק"  
  },  
  "Denver": {  
    "exemplarCity": "דנוור"  
  },  
  "Detroit": {  
    "exemplarCity": "דטרויט"  
  },  
  "Dominica": {
```

```
    "exemplarCity": "דומיניקה"
  },
  "Edmonton": {
    "exemplarCity": "אדמונטון"
  },
  "Eirunepe": {
    "exemplarCity": "אירונפי"
  },
  "El_Salvador": {
    "exemplarCity": "אל סלבדור"
  },
  "Fort_Nelson": {
    "exemplarCity": "פורט נלסון"
  },
  "Fortaleza": {
    "exemplarCity": "פורטאלזה"
  },
  "Glace_Bay": {
    "exemplarCity": "גלייס ביי"
  },
  "Godthab": {
    "exemplarCity": "נואוק"
  },
  "Goose_Bay": {
    "exemplarCity": "גוס ביי"
  },
  "Grand_Turk": {
    "exemplarCity": "גרנד טורק"
  },
  "Grenada": {
    "exemplarCity": "גרנדה"
  },
  "Guadeloupe": {
    "exemplarCity": "גואדלופ"
  },
  "Guatemala": {
    "exemplarCity": "גואטמלה"
  },
  "Guayaquil": {
    "exemplarCity": "גואיאקיל"
  },
  "Guyana": {
    "exemplarCity": "גיאנה"
  },
  "Halifax": {
    "exemplarCity": "הליפקס"
  },
  "Havana": {
    "exemplarCity": "הוואנה"
  },
  "Hermosillo": {
    "exemplarCity": "הרמוסיו"
  },
  "Indiana": {
    "Vincennes": {
      "exemplarCity": "וינסנס, אינדיאנה"
    }
  },
}
```

```
"Petersburg": {
  "exemplarCity": "פיטרסבורג, אינדיאנה"
},
"Tell_City": {
  "exemplarCity": "טל סיטי, אינדיאנה"
},
"Knox": {
  "exemplarCity": "נוקס, אינדיאנה"
},
"Winamac": {
  "exemplarCity": "ווינמאק, אינדיאנה"
},
"Marengo": {
  "exemplarCity": "מרנגו, אינדיאנה"
},
"Vevay": {
  "exemplarCity": "ויוואיי, אינדיאנה"
},
},
"Indianapolis": {
  "exemplarCity": "אינדיאנפוליס"
},
},
"Inuvik": {
  "exemplarCity": "אינוויק"
},
},
"Iqaluit": {
  "exemplarCity": "איקלואיט"
},
},
"Jamaica": {
  "exemplarCity": "ג'מייקה"
},
},
"Jujuy": {
  "exemplarCity": "חוחוי"
},
},
"Juneau": {
  "exemplarCity": "ג'וננו"
},
},
"Kentucky": {
  "Monticello": {
    "exemplarCity": "מונטיצ'לו, קנטאקי"
  }
},
},
"Kralendijk": {
  "exemplarCity": "קרלנדייק"
},
},
"La_Paz": {
  "exemplarCity": "לה פאס"
},
},
"Lima": {
  "exemplarCity": "לימה"
},
},
"Los_Angeles": {
  "exemplarCity": "לוס אנג'לס"
},
},
"Louisville": {
  "exemplarCity": "לואיוויל"
},
},
```

```
"Lower_Princes": {
  "exemplarCity": "לואוור פרינסס קוורטר",
},
"Maceio": {
  "exemplarCity": "מסיאיו",
},
"Managua": {
  "exemplarCity": "מנגואה",
},
"Manaus": {
  "exemplarCity": "מנאוס",
},
"Marigot": {
  "exemplarCity": "מריגו",
},
"Martinique": {
  "exemplarCity": "מרטיניק",
},
"Matamoros": {
  "exemplarCity": "מטמורוס",
},
"Mazatlan": {
  "exemplarCity": "מזטלן",
},
"Mendoza": {
  "exemplarCity": "מנדוזזה",
},
"Menominee": {
  "exemplarCity": "מנומיני",
},
"Merida": {
  "exemplarCity": "מרידה",
},
"Metlakatla": {
  "exemplarCity": "מטלקטלה",
},
"Mexico_City": {
  "exemplarCity": "מקסיקו סיטי",
},
"Miquelon": {
  "exemplarCity": "מיקלון",
},
"Moncton": {
  "exemplarCity": "מונקטון",
},
"Monterrey": {
  "exemplarCity": "מונטריי",
},
"Montevideo": {
  "exemplarCity": "מונטווידאו",
},
"Montserrat": {
  "exemplarCity": "מונטראט",
},
"Nassau": {
  "exemplarCity": "נסאו",
},
},
```



```

    "New_York": {
      "exemplarCity": "ניו יורק"
    },
    "Nipigon": {
      "exemplarCity": "ניפיגון"
    },
    "Nome": {
      "exemplarCity": "נום"
    },
    "Noronha": {
      "exemplarCity": "נורוניה"
    },
    "North_Dakota": {
      "Beulah": {
        "exemplarCity": "צפון דקוטה, ביולה"
      },
      "New_Salem": {
        "exemplarCity": "ניו סייילס, צפון דקוטה"
      },
      "Center": {
        "exemplarCity": "סנטר, צפון דקוטה"
      }
    },
    "Ojinaga": {
      "exemplarCity": "אוג'ינאגה"
    },
    "Panama": {
      "exemplarCity": "פנמה"
    },
    "Pangnirtung": {
      "exemplarCity": "פנגנירטונג"
    },
    "Paramaribo": {
      "exemplarCity": "פרמריבו"
    },
    "Phoenix": {
      "exemplarCity": "פיניקס"
    },
    "Port-au-Prince": {
      "exemplarCity": "פורט או פראנס"
    },
    "Port_of_Spain": {
      "exemplarCity": "פורט אוף ספייין"
    },
    "Porto_Velho": {
      "exemplarCity": "פורטו וליו"
    },
    "Puerto_Rico": {
      "exemplarCity": "פוארטו ריקו"
    },
    "Rainy_River": {
      "exemplarCity": "רייני ריבר"
    },
    "Rankin_Inlet": {
      "exemplarCity": "רנקין אינלט"
    },
    "Recife": {

```

```

    "exemplarCity": "רסיפה"
  },
  "Regina": {
    "exemplarCity": "רג'ינה"
  },
  "Resolute": {
    "exemplarCity": "רזולוט"
  },
  "Rio_Branco": {
    "exemplarCity": "ריו ברנקו"
  },
  "Santa_Isabel": {
    "exemplarCity": "סנטה איסבל"
  },
  "Santarem": {
    "exemplarCity": "סנטרם"
  },
  "Santiago": {
    "exemplarCity": "סנטיאגו"
  },
  "Santo_Domingo": {
    "exemplarCity": "סנטו דומינגו"
  },
  "Sao_Paulo": {
    "exemplarCity": "סאו פאולו"
  },
  "Scoresbysund": {
    "exemplarCity": "סקורסביסונד"
  },
  "Sitka": {
    "exemplarCity": "סיטקה"
  },
  "St_Barthelemy": {
    "exemplarCity": "סנט ברתלמי"
  },
  "St_Johns": {
    "exemplarCity": "סנט ג'ונס"
  },
  "St_Kitts": {
    "exemplarCity": "סנט קיטס"
  },
  "St_Lucia": {
    "exemplarCity": "סנט לוסיה"
  },
  "St_Thomas": {
    "exemplarCity": "סנט תומאס"
  },
  "St_Vincent": {
    "exemplarCity": "סנט וינסנט"
  },
  "Swift_Current": {
    "exemplarCity": "סוויפט קרנט"
  },
  "Tegucigalpa": {
    "exemplarCity": "טגוסיגלפה"
  },
  "Thule": {

```

```

        "exemplarCity": "תולה"
    },
    "Thunder_Bay": {
        "exemplarCity": "ת'אנדר ביי"
    },
    "Tijuana": {
        "exemplarCity": "טיחואנה"
    },
    "Toronto": {
        "exemplarCity": "טורונטו"
    },
    "Tortola": {
        "exemplarCity": "טורטולה"
    },
    "Vancouver": {
        "exemplarCity": "ונקובר"
    },
    "Whitehorse": {
        "exemplarCity": "ווייטהורס"
    },
    "Winnipeg": {
        "exemplarCity": "וויניפג"
    },
    "Yakutat": {
        "exemplarCity": "יקוטאט"
    },
    "Yellowknife": {
        "exemplarCity": "ילונייף"
    }
},
"Atlantic": {
    "Azores": {
        "exemplarCity": "האיים האזוריים"
    },
    "Bermuda": {
        "exemplarCity": "ברמודה"
    },
    "Canary": {
        "exemplarCity": "האיים הקנריים"
    },
    "Cape_Verde": {
        "exemplarCity": "כף ורדה"
    },
    "Faeroe": {
        "exemplarCity": "פארו"
    },
    "Madeira": {
        "exemplarCity": "מדיירה"
    },
    "Reykjavik": {
        "exemplarCity": "רייקיאוויק"
    },
    "South_Georgia": {
        "exemplarCity": "דרום ג'ורג'יה"
    },
    "St_Helena": {
        "exemplarCity": "סנט הלנה"
    }
}

```

```
    },
    "Stanley": {
      "exemplarCity": "סטנלי"
    }
  },
  "Europe": {
    "Amsterdam": {
      "exemplarCity": "אמסטרדם"
    },
    "Andorra": {
      "exemplarCity": "אנדורה"
    },
    "Astrakhan": {
      "exemplarCity": "אסטרחן"
    },
    "Athens": {
      "exemplarCity": "אתונה"
    },
    "Belgrade": {
      "exemplarCity": "בלגרד"
    },
    "Berlin": {
      "exemplarCity": "ברלין"
    },
    "Bratislava": {
      "exemplarCity": "ברטיסלבה"
    },
    "Brussels": {
      "exemplarCity": "בריסל"
    },
    "Bucharest": {
      "exemplarCity": "בוקרשט"
    },
    "Budapest": {
      "exemplarCity": "בודפשט"
    },
    "Busingen": {
      "exemplarCity": "ביזינגן"
    },
    "Chisinau": {
      "exemplarCity": "קיישינב"
    },
    "Copenhagen": {
      "exemplarCity": "קופנהגן"
    },
    "Dublin": {
      "long": {
        "daylight": "שעון קיץ אירלנד"
      },
      "exemplarCity": "דבלין"
    },
    "Gibraltar": {
      "exemplarCity": "גיברלטר"
    },
    "Guernsey": {
      "exemplarCity": "גרנזי"
    }
  },
}
```

```
"Helsinki": {
  "exemplarCity": "הלסינקי",
},
"Isle_of_Man": {
  "exemplarCity": "האי מאן",
},
"Istanbul": {
  "exemplarCity": "איסטנבול",
},
"Jersey": {
  "exemplarCity": "ג'רזי",
},
"Kaliningrad": {
  "exemplarCity": "קלינינגרד",
},
"Kiev": {
  "exemplarCity": "קייב",
},
"Kirov": {
  "exemplarCity": "קירוב",
},
"Lisbon": {
  "exemplarCity": "ליסבון",
},
"Ljubljana": {
  "exemplarCity": "לובליאנה",
},
"London": {
  "long": {
    "daylight": "שעון קיץ בריטניה",
  },
  "exemplarCity": "לונדון",
},
"Luxembourg": {
  "exemplarCity": "לוקסמבורג",
},
"Madrid": {
  "exemplarCity": "מדריד",
},
"Malta": {
  "exemplarCity": "מלטה",
},
"Mariehamn": {
  "exemplarCity": "מריהאמן",
},
"Minsk": {
  "exemplarCity": "מינסק",
},
"Monaco": {
  "exemplarCity": "מונקו",
},
"Moscow": {
  "exemplarCity": "מוסקבה",
},
"Oslo": {
  "exemplarCity": "אוסלו",
},
},
```

```
"Paris": {
  "exemplarCity": "פריז"
},
"Podgorica": {
  "exemplarCity": "פודגוריצה"
},
"Prague": {
  "exemplarCity": "פראג"
},
"Riga": {
  "exemplarCity": "ריגה"
},
"Rome": {
  "exemplarCity": "רומא"
},
"Samara": {
  "exemplarCity": "סמרה"
},
"San_Marino": {
  "exemplarCity": "סן מרינו"
},
"Sarajevo": {
  "exemplarCity": "סרייבו"
},
"Simferopol": {
  "exemplarCity": "סימפרופול"
},
"Skopje": {
  "exemplarCity": "סקופיה"
},
"Sofia": {
  "exemplarCity": "סופיה"
},
"Stockholm": {
  "exemplarCity": "סטוקהולם"
},
"Tallinn": {
  "exemplarCity": "טאלין"
},
"Tirane": {
  "exemplarCity": "טירנה"
},
"Ulyanovsk": {
  "exemplarCity": "אוליאנובסק"
},
"Uzhgorod": {
  "exemplarCity": "אוז'הורוד"
},
"Vaduz": {
  "exemplarCity": "ואדוץ"
},
"Vatican": {
  "exemplarCity": "הוותיקן"
},
"Vienna": {
  "exemplarCity": "וינה"
},
},
```

```
"Vilnius": {
  "exemplarCity": "וילנה"
},
"Volgograd": {
  "exemplarCity": "וולגוגרד"
},
"Warsaw": {
  "exemplarCity": "ורשה"
},
"Zagreb": {
  "exemplarCity": "זאגרב"
},
"Zaporozhye": {
  "exemplarCity": "זפורוז'יה"
},
"Zurich": {
  "exemplarCity": "ציריך"
}
},
"Africa": {
  "Abidjan": {
    "exemplarCity": "אביג'אן"
  },
  "Accra": {
    "exemplarCity": "אקרה"
  },
  "Addis_Ababa": {
    "exemplarCity": "אדיס אבבה"
  },
  "Algiers": {
    "exemplarCity": "אלג'יר"
  },
  "Asmera": {
    "exemplarCity": "אסמרה"
  },
  "Bamako": {
    "exemplarCity": "במאקו"
  },
  "Bangui": {
    "exemplarCity": "בנגואי"
  },
  "Banjul": {
    "exemplarCity": "בנג'ול"
  },
  "Bissau": {
    "exemplarCity": "ביסאו"
  },
  "Blantyre": {
    "exemplarCity": "בלנטיר"
  },
  "Brazzaville": {
    "exemplarCity": "ברזוויל"
  },
  "Bujumbura": {
    "exemplarCity": "בוג'ומבורה"
  },
  "Cairo": {
```

```
    "exemplarCity": "קהיר"  
  },  
  "Casablanca": {  
    "exemplarCity": "קזבלנקה"  
  },  
  "Ceuta": {  
    "exemplarCity": "סאוטה"  
  },  
  "Conakry": {  
    "exemplarCity": "קונאקרי"  
  },  
  "Dakar": {  
    "exemplarCity": "דקאר"  
  },  
  "Dar_es_Salaam": {  
    "exemplarCity": "דאר א-סלאם"  
  },  
  "Djibouti": {  
    "exemplarCity": "ג'יבוטי"  
  },  
  "Douala": {  
    "exemplarCity": "דואלה"  
  },  
  "El_Aaiun": {  
    "exemplarCity": "אל עיון"  
  },  
  "Freetown": {  
    "exemplarCity": "פריטאון"  
  },  
  "Gaborone": {  
    "exemplarCity": "גבורונה"  
  },  
  "Harare": {  
    "exemplarCity": "הרארה"  
  },  
  "Johannesburg": {  
    "exemplarCity": "יוהנסבורג"  
  },  
  "Juba": {  
    "exemplarCity": "ג'ובה"  
  },  
  "Kampala": {  
    "exemplarCity": "קמפלה"  
  },  
  "Khartoum": {  
    "exemplarCity": "חרטום"  
  },  
  "Kigali": {  
    "exemplarCity": "קיגלי"  
  },  
  "Kinshasa": {  
    "exemplarCity": "קינשה"  
  },  
  "Lagos": {  
    "exemplarCity": "לגוס"  
  },  
  "Libreville": {
```



```
    "exemplarCity": "ליברוויל"
  },
  "Lome": {
    "exemplarCity": "לומה"
  },
  "Luanda": {
    "exemplarCity": "לואנדה"
  },
  "Lubumbashi": {
    "exemplarCity": "לובומבאשי"
  },
  "Lusaka": {
    "exemplarCity": "לוסקה"
  },
  "Malabo": {
    "exemplarCity": "מלבו"
  },
  "Maputo": {
    "exemplarCity": "מאפוטו"
  },
  "Maseru": {
    "exemplarCity": "מסרו"
  },
  "Mbabane": {
    "exemplarCity": "אמבאבאנה"
  },
  "Mogadishu": {
    "exemplarCity": "מוגדישו"
  },
  "Monrovia": {
    "exemplarCity": "מונרוביה"
  },
  "Nairobi": {
    "exemplarCity": "ניירובי"
  },
  "Ndjamena": {
    "exemplarCity": "נג'מנה"
  },
  "Niamey": {
    "exemplarCity": "ניאמי"
  },
  "Nouakchott": {
    "exemplarCity": "נואקצ'וט"
  },
  "Ouagadougou": {
    "exemplarCity": "וואגאדוגו"
  },
  "Porto-Novo": {
    "exemplarCity": "פורטו נובו"
  },
  "Sao_Tome": {
    "exemplarCity": "סאו טומה"
  },
  "Tripoli": {
    "exemplarCity": "טריפולי"
  },
  "Tunis": {
```

```

        "exemplarCity": "תוניס"
      },
      "Windhoek": {
        "exemplarCity": "ווינדהוק"
      }
    },
    "Asia": {
      "Aden": {
        "exemplarCity": "עדן"
      },
      "Almaty": {
        "exemplarCity": "אלמאטי"
      },
      "Amman": {
        "exemplarCity": "עמאן"
      },
      "Anadyr": {
        "exemplarCity": "אנדיר"
      },
      "Aqtau": {
        "exemplarCity": "אקטאון"
      },
      "Aqtobe": {
        "exemplarCity": "אקטובה"
      },
      "Ashgabat": {
        "exemplarCity": "אשגבט"
      },
      "Baghdad": {
        "exemplarCity": "בגדד"
      },
      "Bahrain": {
        "exemplarCity": "בחריין"
      },
      "Baku": {
        "exemplarCity": "באקו"
      },
      "Bangkok": {
        "exemplarCity": "בנגקוק"
      },
      "Barnaul": {
        "exemplarCity": "ברנאול"
      },
      "Beirut": {
        "exemplarCity": "ביירות"
      },
      "Bishkek": {
        "exemplarCity": "בישקק"
      },
      "Brunei": {
        "exemplarCity": "ברוניי"
      },
      "Calcutta": {
        "exemplarCity": "קולקטה"
      },
      "Chita": {
        "exemplarCity": "צ'יטה"
      }
    }
  }

```

```
    },  
    "Choibalsan": {  
      "exemplarCity": "צ'ויבלסן"  
    },  
    "Colombo": {  
      "exemplarCity": "קולומבו"  
    },  
    "Damascus": {  
      "exemplarCity": "דמשק"  
    },  
    "Dhaka": {  
      "exemplarCity": "דאקה"  
    },  
    "Dili": {  
      "exemplarCity": "דילי"  
    },  
    "Dubai": {  
      "exemplarCity": "דובאי"  
    },  
    "Dushanbe": {  
      "exemplarCity": "דושנבה"  
    },  
    "Gaza": {  
      "exemplarCity": "עזה"  
    },  
    "Hebron": {  
      "exemplarCity": "חברון"  
    },  
    "Hong_Kong": {  
      "exemplarCity": "הונג קונג"  
    },  
    "Hovd": {  
      "exemplarCity": "חובד"  
    },  
    "Irkutsk": {  
      "exemplarCity": "אירקוטסק"  
    },  
    "Jakarta": {  
      "exemplarCity": "ג'קרטה"  
    },  
    "Jayapura": {  
      "exemplarCity": "ג'איאפורה"  
    },  
    "Jerusalem": {  
      "exemplarCity": "ירושלים"  
    },  
    "Kabul": {  
      "exemplarCity": "קאבול"  
    },  
    "Kamchatka": {  
      "exemplarCity": "קמצ'טקה"  
    },  
    "Karachi": {  
      "exemplarCity": "קראצ'י"  
    },  
    "Katmandu": {  
      "exemplarCity": "קטמנדו"
```

```
    },
    "Khandyga": {
      "exemplarCity": "חנדיגה"
    },
    "Krasnoyarsk": {
      "exemplarCity": "קרסנויארסק"
    },
    "Kuala_Lumpur": {
      "exemplarCity": "קואלה לומפור"
    },
    "Kuching": {
      "exemplarCity": "קוצ'ינג"
    },
    "Kuwait": {
      "exemplarCity": "כווית"
    },
    "Macau": {
      "exemplarCity": "מקאו"
    },
    "Magadan": {
      "exemplarCity": "מגדן"
    },
    "Makassar": {
      "exemplarCity": "מאקאסאר"
    },
    "Manila": {
      "exemplarCity": "מנילה"
    },
    "Muscat": {
      "exemplarCity": "מוסקט"
    },
    "Nicosia": {
      "exemplarCity": "ניקוסיה"
    },
    "Novokuznetsk": {
      "exemplarCity": "נובוקוזנטסק"
    },
    "Novosibirsk": {
      "exemplarCity": "נובוסירסק"
    },
    "Omsk": {
      "exemplarCity": "אומסק"
    },
    "Oral": {
      "exemplarCity": "אורל"
    },
    "Phnom_Penh": {
      "exemplarCity": "פנום פן"
    },
    "Pontianak": {
      "exemplarCity": "פונטיאנק"
    },
    "Pyongyang": {
      "exemplarCity": "פיונגיאנג"
    },
    "Qatar": {
      "exemplarCity": "קטאר"
```

```
    },
    "Qyzylorda": {
      "exemplarCity": "קִיזִילֹרְדָה"
    },
    "Rangoon": {
      "exemplarCity": "רַנְגּוֹן"
    },
    "Riyadh": {
      "exemplarCity": "רִיאַד"
    },
    "Saigon": {
      "exemplarCity": "הוֹ צִי מִין סִיטִי"
    },
    "Sakhalin": {
      "exemplarCity": "סַחְלִין"
    },
    "Samarkand": {
      "exemplarCity": "סַמַרְקַנְד"
    },
    "Seoul": {
      "exemplarCity": "סֵאוּל"
    },
    "Shanghai": {
      "exemplarCity": "שַׁנְחַאִי"
    },
    "Singapore": {
      "exemplarCity": "סִינְגַפּוּר"
    },
    "Srednekolymsk": {
      "exemplarCity": "סֶרְדֶנִיקוֹלִימְסַק"
    },
    "Taipei": {
      "exemplarCity": "טַאִיפֵי"
    },
    "Tashkent": {
      "exemplarCity": "טַשְׁקֶנְט"
    },
    "Tbilisi": {
      "exemplarCity": "טְבִילִיסִי"
    },
    "Tehran": {
      "exemplarCity": "טֶהְרַן"
    },
    "Thimphu": {
      "exemplarCity": "טֶהִימְפֹהוּ"
    },
    "Tokyo": {
      "exemplarCity": "טוֹקִיו"
    },
    "Tomsk": {
      "exemplarCity": "טוֹמְסַק"
    },
    "Ulaanbaatar": {
      "exemplarCity": "אוּלַאנְבַטַאר"
    },
    "Urumqi": {
      "exemplarCity": "אוּרוּמְקִי"
    }
  }
```

```
    },
    "Ust-Nera": {
      "exemplarCity": "אוסט-נרה"
    },
    "Vientiane": {
      "exemplarCity": "האנוי"
    },
    "Vladivostok": {
      "exemplarCity": "ולדיוווסטוק"
    },
    "Yakutsk": {
      "exemplarCity": "יקוטסק"
    },
    "Yekaterinburg": {
      "exemplarCity": "יקטרינבורג"
    },
    "Yerevan": {
      "exemplarCity": "ירוואן"
    }
  },
  "Indian": {
    "Antananarivo": {
      "exemplarCity": "אנטננריבו"
    },
    "Chagos": {
      "exemplarCity": "צ'אגוס"
    },
    "Christmas": {
      "exemplarCity": "האי כריסטמס"
    },
    "Cocos": {
      "exemplarCity": "קוקוס"
    },
    "Comoro": {
      "exemplarCity": "קומורו"
    },
    "Kerguelen": {
      "exemplarCity": "קרגוולן"
    },
    "Mahe": {
      "exemplarCity": "מהא"
    },
    "Maldives": {
      "exemplarCity": "האיים המלדיביים"
    },
    "Mauritius": {
      "exemplarCity": "מאוריציוס"
    },
    "Mayotte": {
      "exemplarCity": "מאיוט"
    },
    "Reunion": {
      "exemplarCity": "ראוניון"
    }
  },
  "Australia": {
    "Adelaide": {
```

```

        "exemplarCity": "אדלייד"
      },
      "Brisbane": {
        "exemplarCity": "בריסביין"
      },
      "Broken_Hill": {
        "exemplarCity": "ברוקן היל"
      },
      "Currie": {
        "exemplarCity": "קרי"
      },
      "Darwin": {
        "exemplarCity": "דרווין"
      },
      "Eucla": {
        "exemplarCity": "יוקלה"
      },
      "Hobart": {
        "exemplarCity": "הוברט"
      },
      "Lindeman": {
        "exemplarCity": "לינדמן"
      },
      "Lord_Howe": {
        "exemplarCity": "אי הלורד האו"
      },
      "Melbourne": {
        "exemplarCity": "מלבורן"
      },
      "Perth": {
        "exemplarCity": "פרת'
      },
      "Sydney": {
        "exemplarCity": "סידני"
      }
    },
    "Pacific": {
      "Apia": {
        "exemplarCity": "אפיה"
      },
      "Auckland": {
        "exemplarCity": "אוקלנד"
      },
      "Bougainville": {
        "exemplarCity": "בוגנוויל"
      },
      "Chatham": {
        "exemplarCity": "צ'אטהאם"
      },
      "Easter": {
        "exemplarCity": "אי הפסחא"
      },
      "Efate": {
        "exemplarCity": "אפטה"
      },
      "Enderbury": {
        "exemplarCity": "אנדרבורי"
      }
    }
  }

```

```
    },
    "Fakaofo": {
      "exemplarCity": "פקאופו"
    },
    "Fiji": {
      "exemplarCity": "פיג'י"
    },
    "Funafuti": {
      "exemplarCity": "פונפוט"
    },
    "Galapagos": {
      "exemplarCity": "גלפאגוס"
    },
    "Gambier": {
      "exemplarCity": "איי גמבייה"
    },
    "Guadalcanal": {
      "exemplarCity": "גוודלקנאל"
    },
    "Guam": {
      "exemplarCity": "גואם"
    },
    "Honolulu": {
      "exemplarCity": "הונוולו"
    },
    "Johnston": {
      "exemplarCity": "ג'ונסטון"
    },
    "Kiritimati": {
      "exemplarCity": "קיריטימאטי"
    },
    "Kosrae": {
      "exemplarCity": "קוסרה"
    },
    "Kwajalein": {
      "exemplarCity": "קוואג'ליין"
    },
    "Majuro": {
      "exemplarCity": "מאג'ורו"
    },
    "Marquesas": {
      "exemplarCity": "איי מרקז"
    },
    "Midway": {
      "exemplarCity": "מידוויי"
    },
    "Nauru": {
      "exemplarCity": "נאורו"
    },
    "Niue": {
      "exemplarCity": "ניואה"
    },
    "Norfolk": {
      "exemplarCity": "נורפוק"
    },
    "Noumea": {
      "exemplarCity": "נומאה"
    }
  }
```



```

    },
    "Pago_Pago": {
      "exemplarCity": "פאגו פאגו"
    },
    "Palau": {
      "exemplarCity": "פלאו"
    },
    "Pitcairn": {
      "exemplarCity": "פיטקרן"
    },
    "Ponape": {
      "exemplarCity": "פונפיי"
    },
    "Port_Moresby": {
      "exemplarCity": "פורט מורסבי"
    },
    "Rarotonga": {
      "exemplarCity": "רארוטונגה"
    },
    "Saipan": {
      "exemplarCity": "סאיפאן"
    },
    "Tahiti": {
      "exemplarCity": "טהיטי"
    },
    "Tarawa": {
      "exemplarCity": "טאראווה"
    },
    "Tongatapu": {
      "exemplarCity": "טונגטאפו"
    },
    "Truk": {
      "exemplarCity": "צ'וק"
    },
    "Wake": {
      "exemplarCity": "וויק"
    },
    "Wallis": {
      "exemplarCity": "ווליס"
    }
  },
  "Arctic": {
    "Longyearbyen": {
      "exemplarCity": "לונגיירבין"
    }
  },
  "Antarctica": {
    "Casey": {
      "exemplarCity": "קאסיי"
    },
    "Davis": {
      "exemplarCity": "דייוויס"
    },
    "DumontDUrville": {
      "exemplarCity": "דומון ד'אורוויל"
    },
    "Macquarie": {

```

```

        "exemplarCity": "מקריי"
      },
      "Mawson": {
        "exemplarCity": "מוסטון"
      },
      "McMurdo": {
        "exemplarCity": "מק-מרדו"
      },
      "Palmer": {
        "exemplarCity": "פאלמר"
      },
      "Rothera": {
        "exemplarCity": "רות'רה"
      },
      "Syowa": {
        "exemplarCity": "סיוואה"
      },
      "Troll": {
        "exemplarCity": "טרול"
      },
      "Vostok": {
        "exemplarCity": "ווסטוק"
      }
    },
    "Etc": {
      "GMT": {
        "exemplarCity": "GMT"
      },
      "GMT1": {
        "exemplarCity": "GMT+1"
      },
      "GMT10": {
        "exemplarCity": "GMT+10"
      },
      "GMT11": {
        "exemplarCity": "GMT+11"
      },
      "GMT12": {
        "exemplarCity": "GMT+12"
      },
      "GMT2": {
        "exemplarCity": "GMT+2"
      },
      "GMT3": {
        "exemplarCity": "GMT+3"
      },
      "GMT4": {
        "exemplarCity": "GMT+4"
      },
      "GMT5": {
        "exemplarCity": "GMT+5"
      },
      "GMT6": {
        "exemplarCity": "GMT+6"
      },
      "GMT7": {
        "exemplarCity": "GMT+7"
      }
    }
  }

```

```
    },
    "GMT8": {
      "exemplarCity": "GMT+8"
    },
    "GMT9": {
      "exemplarCity": "GMT+9"
    },
    "GMT-1": {
      "exemplarCity": "GMT-1"
    },
    "GMT-10": {
      "exemplarCity": "GMT-10"
    },
    "GMT-11": {
      "exemplarCity": "GMT-11"
    },
    "GMT-12": {
      "exemplarCity": "GMT-12"
    },
    "GMT-13": {
      "exemplarCity": "GMT-13"
    },
    "GMT-14": {
      "exemplarCity": "GMT-14"
    },
    "GMT-2": {
      "exemplarCity": "GMT-2"
    },
    "GMT-3": {
      "exemplarCity": "GMT-3"
    },
    "GMT-4": {
      "exemplarCity": "GMT-4"
    },
    "GMT-5": {
      "exemplarCity": "GMT-5"
    },
    "GMT-6": {
      "exemplarCity": "GMT-6"
    },
    "GMT-7": {
      "exemplarCity": "GMT-7"
    },
    "GMT-8": {
      "exemplarCity": "GMT-8"
    },
    "GMT-9": {
      "exemplarCity": "GMT-9"
    },
    "UTC": {
      "long": {
        "standard": "זמן אוניברסלי מתואם"
      },
      "short": {
        "standard": "UTC"
      },
      "exemplarCity": "UTC"
    }
  },
```

```

    },
    "Unknown": {
      "exemplarCity": "עיר לא ידועה"
    }
  },
  "metazone": {
    "Afghanistan": {
      "long": {
        "standard": "שעון אפגניסטן"
      }
    },
    "Africa_Central": {
      "long": {
        "standard": "שעון מרכז אפריקה"
      }
    },
    "Africa_Eastern": {
      "long": {
        "standard": "שעון מזרח אפריקה"
      }
    },
    "Africa_Southern": {
      "long": {
        "standard": "שעון דרום אפריקה"
      }
    },
    "Africa_Western": {
      "long": {
        "generic": "שעון מערב אפריקה",
        "standard": "שעון מערב אפריקה (חורף)",
        "daylight": "שעון מערב אפריקה (קיץ)"
      }
    },
    "Alaska": {
      "long": {
        "generic": "שעון אלסקה",
        "standard": "שעון אלסקה (חורף)",
        "daylight": "שעון אלסקה (קיץ)"
      }
    },
    "Amazon": {
      "long": {
        "generic": "שעון אמזונס",
        "standard": "שעון אמזונס (חורף)",
        "daylight": "שעון אמזונס (קיץ)"
      }
    },
    "America_Central": {
      "long": {
        "generic": "שעון מרכז ארה"ב",
        "standard": "שעון מרכז ארה"ב (חורף)",
        "daylight": "שעון מרכז ארה"ב (קיץ)"
      }
    },
    "America_Eastern": {
      "long": {

```

```

        "generic": "שעון החוף המזרחי",
        "standard": "שעון החוף המזרחי (חורף)",
        "daylight": "שעון החוף המזרחי (קיץ)"
    },
},
"America_Mountain": {
    "long": {
        "generic": "שעון אזור ההרים בארה"ב",
        "standard": "שעון אזור ההרים בארה"ב (חורף)",
        "daylight": "שעון אזור ההרים בארה"ב (קיץ)"
    }
},
"America_Pacific": {
    "long": {
        "generic": "שעון מערב ארה"ב",
        "standard": "שעון מערב ארה"ב (חורף)",
        "daylight": "שעון מערב ארה"ב (קיץ)"
    }
},
"Anadyr": {
    "long": {
        "generic": "שעון אנדיר",
        "standard": "שעון רגיל אנדיר",
        "daylight": "שעון קיץ אנדיר"
    }
},
"Apia": {
    "long": {
        "generic": "שעון אפיה",
        "standard": "שעון אפיה (חורף)",
        "daylight": "שעון אפיה (קיץ)"
    }
},
"Arabian": {
    "long": {
        "generic": "שעון חצי האי ערב",
        "standard": "שעון חצי האי ערב (חורף)",
        "daylight": "שעון חצי האי ערב (קיץ)"
    }
},
"Argentina": {
    "long": {
        "generic": "שעון ארגנטינה",
        "standard": "שעון ארגנטינה (חורף)",
        "daylight": "שעון ארגנטינה (קיץ)"
    }
},
"Argentina_Western": {
    "long": {
        "generic": "שעון מערב ארגנטינה",
        "standard": "שעון מערב ארגנטינה (חורף)",
        "daylight": "שעון מערב ארגנטינה (קיץ)"
    }
},
"Armenia": {
    "long": {
        "generic": "שעון ארמניה",

```

```

        "standard": "שעון ארמניה (חורף)",
        "daylight": "שעון ארמניה (קיץ)"
    },
    },
    "Atlantic": {
        "long": {
            "generic": "שעון האוקיינוס האטלנטי",
            "standard": "שעון האוקיינוס האטלנטי (חורף)",
            "daylight": "שעון האוקיינוס האטלנטי (קיץ)"
        }
    },
    "Australia_Central": {
        "long": {
            "generic": "שעון מרכז אוסטרליה",
            "standard": "שעון מרכז אוסטרליה (חורף)",
            "daylight": "שעון מרכז אוסטרליה (קיץ)"
        }
    },
    "Australia_CentralWestern": {
        "long": {
            "generic": "שעון מרכז-מערב אוסטרליה",
            "standard": "שעון מרכז-מערב אוסטרליה (חורף)",
            "daylight": "שעון מרכז-מערב אוסטרליה (קיץ)"
        }
    },
    "Australia_Eastern": {
        "long": {
            "generic": "שעון מזרח אוסטרליה",
            "standard": "שעון מזרח אוסטרליה (חורף)",
            "daylight": "שעון מזרח אוסטרליה (קיץ)"
        }
    },
    "Australia_Western": {
        "long": {
            "generic": "שעון מערב אוסטרליה",
            "standard": "שעון מערב אוסטרליה (חורף)",
            "daylight": "שעון מערב אוסטרליה (קיץ)"
        }
    },
    "Azerbaijan": {
        "long": {
            "generic": "שעון אזרבייג'אן",
            "standard": "שעון אזרבייג'אן (חורף)",
            "daylight": "שעון אזרבייג'אן (קיץ)"
        }
    },
    "Azores": {
        "long": {
            "generic": "שעון האיים האזוריים",
            "standard": "שעון האיים האזוריים (חורף)",
            "daylight": "שעון האיים האזוריים (קיץ)"
        }
    },
    "Bangladesh": {
        "long": {
            "generic": "שעון בנגלדש",
            "standard": "שעון בנגלדש (חורף)",

```

```

        "daylight": "שעון בנגלדש (קייץ)"
    },
    },
    "Bhutan": {
        "long": {
            "standard": "שעון בהוטן"
        }
    },
    },
    "Bolivia": {
        "long": {
            "standard": "שעון בוליביה"
        }
    },
    },
    "Brasilia": {
        "long": {
            "generic": "שעון ברזיליה",
            "standard": "שעון ברזיליה (חורף)",
            "daylight": "שעון ברזיליה (קייץ)"
        }
    },
    },
    "Brunei": {
        "long": {
            "standard": "שעון ברוניי דארוסלאם"
        }
    },
    },
    "Cape_Verde": {
        "long": {
            "generic": "שעון כף ורדה",
            "standard": "שעון כף ורדה (חורף)",
            "daylight": "שעון כף ורדה (קייץ)"
        }
    },
    },
    "Chamorro": {
        "long": {
            "standard": "שעון צ'אמורו"
        }
    },
    },
    "Chatham": {
        "long": {
            "generic": "שעון צ'טהאם",
            "standard": "שעון צ'טהאם (חורף)",
            "daylight": "שעון צ'טהאם (קייץ)"
        }
    },
    },
    "Chile": {
        "long": {
            "generic": "שעון צ'ילה",
            "standard": "שעון צ'ילה (חורף)",
            "daylight": "שעון צ'ילה (קייץ)"
        }
    },
    },
    "China": {
        "long": {
            "generic": "שעון סין",
            "standard": "שעון סין (חורף)",
            "daylight": "שעון סין (קייץ)"
        }
    }
}

```

```

    },
    "Choibalsan": {
      "long": {
        "generic": "שעון צ'ויבלסן",
        "standard": "שעון צ'ויבלסן (חורף)",
        "daylight": "שעון צ'ויבלסן (קיץ)"
      }
    },
    "Christmas": {
      "long": {
        "standard": "שעון האי כריסטמס"
      }
    },
    "Cocos": {
      "long": {
        "standard": "שעון איי קוקוס"
      }
    },
    "Colombia": {
      "long": {
        "generic": "שעון קולומביה",
        "standard": "שעון קולומביה (חורף)",
        "daylight": "שעון קולומביה (קיץ)"
      }
    },
    "Cook": {
      "long": {
        "generic": "שעון איי קוק",
        "standard": "שעון איי קוק (חורף)",
        "daylight": "שעון איי קוק (מחצית הקיץ)"
      }
    },
    "Cuba": {
      "long": {
        "generic": "שעון קובה",
        "standard": "שעון קובה (חורף)",
        "daylight": "שעון קובה (קיץ)"
      }
    },
    "Davis": {
      "long": {
        "standard": "שעון דייוויס"
      }
    },
    "DumontDUrville": {
      "long": {
        "standard": "שעון דומון ד'אורוויל"
      }
    },
    "East_Timor": {
      "long": {
        "standard": "שעון מזרח טימור"
      }
    },
    "Easter": {
      "long": {
        "generic": "שעון אי הפסחא",

```



```

        "standard": "שעון אי הפסחא (חורף)",
        "daylight": "שעון אי הפסחא (קיץ)"
    },
},
"Ecuador": {
    "long": {
        "standard": "שעון אקוודור"
    }
},
"Europe_Central": {
    "long": {
        "generic": "שעון מרכז אירופה",
        "standard": "שעון מרכז אירופה (חורף)",
        "daylight": "שעון מרכז אירופה (קיץ)"
    }
},
"Europe_Eastern": {
    "long": {
        "generic": "שעון מזרח אירופה",
        "standard": "שעון מזרח אירופה (חורף)",
        "daylight": "שעון מזרח אירופה (קיץ)"
    }
},
"Europe_Further_Eastern": {
    "long": {
        "standard": "שעון מינסק"
    }
},
"Europe_Western": {
    "long": {
        "generic": "שעון מערב אירופה",
        "standard": "שעון מערב אירופה (חורף)",
        "daylight": "שעון מערב אירופה (קיץ)"
    }
},
"Falkland": {
    "long": {
        "generic": "שעון איי פוקלנד",
        "standard": "שעון איי פוקלנד (חורף)",
        "daylight": "שעון איי פוקלנד (קיץ)"
    }
},
"Fiji": {
    "long": {
        "generic": "שעון פיג'י",
        "standard": "שעון פיג'י (חורף)",
        "daylight": "שעון פיג'י (קיץ)"
    }
},
"French_Guiana": {
    "long": {
        "standard": "שעון גיאנה הצרפתית"
    }
},
"French_Southern": {
    "long": {
        "standard": "שעון הארצות הדרומיות והאנטארקטיות של צרפת"
    }
}

```

```
    },
    "Galapagos": {
      "long": {
        "standard": "שעון איי גלאפגוס"
      }
    },
    "Gambier": {
      "long": {
        "standard": "שעון איי גמבייה"
      }
    },
    "Georgia": {
      "long": {
        "generic": "שעון גאורגיה",
        "standard": "שעון גאורגיה (חורף)",
        "daylight": "שעון גאורגיה (קיץ)"
      }
    },
    "Gilbert_Islands": {
      "long": {
        "standard": "שעון איי גילברט"
      }
    },
    "GMT": {
      "long": {
        "standard": "שעון גריניץ'"
      }
    },
    "Greenland_Eastern": {
      "long": {
        "generic": "שעון מזרח גרינלנד",
        "standard": "שעון מזרח גרינלנד (חורף)",
        "daylight": "שעון מזרח גרינלנד (קיץ)"
      }
    },
    "Greenland_Western": {
      "long": {
        "generic": "שעון מערב גרינלנד",
        "standard": "שעון מערב גרינלנד (חורף)",
        "daylight": "שעון מערב גרינלנד (קיץ)"
      }
    },
    "Gulf": {
      "long": {
        "standard": "שעון מדינות המפרץ"
      }
    },
    "Guyana": {
      "long": {
        "standard": "שעון גיאנה"
      }
    },
    "Hawaii_Aleutian": {
      "long": {
        "generic": "שעון האיים האלאוטיים הוואי",
        "standard": "שעון האיים האלאוטיים הוואי (חורף)",
```

```

        "daylight": "שעון האיים האלאוטיים הוואי (קיץ)"
    },
    },
    "Hong_Kong": {
        "long": {
            "generic": "שעון הונג קונג",
            "standard": "שעון הונג קונג (חורף)",
            "daylight": "שעון הונג קונג (קיץ)"
        }
    },
    },
    "Hovd": {
        "long": {
            "generic": "שעון חובד",
            "standard": "שעון חובד (חורף)",
            "daylight": "שעון חובד (קיץ)"
        }
    },
    },
    "India": {
        "long": {
            "standard": "שעון הודו"
        }
    },
    },
    "Indian_Ocean": {
        "long": {
            "standard": "שעון האוקיינוס ההודי"
        }
    },
    },
    "Indochina": {
        "long": {
            "standard": "שעון הודו-סין"
        }
    },
    },
    "Indonesia_Central": {
        "long": {
            "standard": "שעון מרכז אינדונזיה"
        }
    },
    },
    "Indonesia_Eastern": {
        "long": {
            "standard": "שעון מזרח אינדונזיה"
        }
    },
    },
    "Indonesia_Western": {
        "long": {
            "standard": "שעון מערב אינדונזיה"
        }
    },
    },
    "Iran": {
        "long": {
            "generic": "שעון איראן",
            "standard": "שעון איראן (חורף)",
            "daylight": "שעון איראן (קיץ)"
        }
    },
    },
    "Irkutsk": {
        "long": {
            "generic": "שעון אירקוטסק",

```

```

        "standard": "שעון אירקוטסק (חורף)",
        "daylight": "שעון אירקוטסק (קיץ)"
    },
},
"Israel": {
    "long": {
        "generic": "שעון ישראל",
        "standard": "שעון ישראל (חורף)",
        "daylight": "שעון ישראל (קיץ)"
    }
},
"Japan": {
    "long": {
        "generic": "שעון יפן",
        "standard": "שעון יפן (חורף)",
        "daylight": "שעון יפן (קיץ)"
    }
},
"Kamchatka": {
    "long": {
        "generic": "שעון פטרופבובסק-קמצ'טסקי",
        "standard": "שעון רגיל פטרופבובסק-קמצ'טסקי",
        "daylight": "שעון קיץ פטרופבובסק-קמצ'טסקי"
    }
},
"Kazakhstan_Eastern": {
    "long": {
        "standard": "שעון מזרח קזחסטן"
    }
},
"Kazakhstan_Western": {
    "long": {
        "standard": "שעון מערב קזחסטן"
    }
},
"Korea": {
    "long": {
        "generic": "שעון קוריאה",
        "standard": "שעון קוריאה (חורף)",
        "daylight": "שעון קוריאה (קיץ)"
    }
},
"Kosrae": {
    "long": {
        "standard": "שעון קוסראה"
    }
},
"Krasnoyarsk": {
    "long": {
        "generic": "שעון קרסנויארסק",
        "standard": "שעון קרסנויארסק (חורף)",
        "daylight": "שעון קרסנויארסק (קיץ)"
    }
},
"Kyrgystan": {
    "long": {
        "standard": "שעון קירגיזסטן"
    }
}

```

```

    },
    "Line_Islands": {
      "long": {
        "standard": "שעון איי ליין"
      }
    },
    "Lord_Howe": {
      "long": {
        "generic": "שעון אי הלורד האו",
        "standard": "שעון אי הלורד האו (חורף)",
        "daylight": "שעון אי הלורד האו (קיץ)"
      }
    },
    "Macau": {
      "long": {
        "generic": "שעון מקאו",
        "standard": "שעון חורף מקאו",
        "daylight": "שעון קיץ מקאו"
      }
    },
    "Macquarie": {
      "long": {
        "standard": "שעון מקווארי"
      }
    },
    "Magadan": {
      "long": {
        "generic": "שעון מגדן",
        "standard": "שעון מגדן (חורף)",
        "daylight": "שעון מגדן (קיץ)"
      }
    },
    "Malaysia": {
      "long": {
        "standard": "שעון מלזיה"
      }
    },
    "Maldives": {
      "long": {
        "standard": "שעון האיים המלדיביים"
      }
    },
    "Marquesas": {
      "long": {
        "standard": "שעון איי מרקז"
      }
    },
    "Marshall_Islands": {
      "long": {
        "standard": "שעון איי מרשל"
      }
    },
    "Mauritius": {
      "long": {
        "generic": "שעון מאוריציוס",
        "standard": "שעון מאוריציוס (חורף)",

```

```

        "daylight": "שעון מאוריצייוס (קייץ)"
    },
    },
    "Mawson": {
        "long": {
            "standard": "שעון מאוסון"
        }
    },
    },
    "Mexico_Northwest": {
        "long": {
            "generic": "שעון צפון-מערב מקסיקו",
            "standard": "שעון צפון-מערב מקסיקו (חורף)",
            "daylight": "שעון צפון-מערב מקסיקו (קייץ)"
        }
    },
    },
    "Mexico_Pacific": {
        "long": {
            "generic": "שעון מערב מקסיקו",
            "standard": "שעון מערב מקסיקו (חורף)",
            "daylight": "שעון מערב מקסיקו (קייץ)"
        }
    },
    },
    "Mongolia": {
        "long": {
            "generic": "שעון אולן בטור",
            "standard": "שעון אולן בטור (חורף)",
            "daylight": "שעון אולן בטור (קייץ)"
        }
    },
    },
    "Moscow": {
        "long": {
            "generic": "שעון מוסקבה",
            "standard": "שעון מוסקבה (חורף)",
            "daylight": "שעון מוסקבה (קייץ)"
        }
    },
    },
    "Myanmar": {
        "long": {
            "standard": "שעון מיאנמר"
        }
    },
    },
    "Nauru": {
        "long": {
            "standard": "שעון נאורו"
        }
    },
    },
    "Nepal": {
        "long": {
            "standard": "שעון נפאל"
        }
    },
    },
    "New_Caledonia": {
        "long": {
            "generic": "שעון קלדוניה החדשה",
            "standard": "שעון קלדוניה החדשה (חורף)",
            "daylight": "שעון קלדוניה החדשה (קייץ)"
        }
    }
}

```

```

    },
    "New_Zealand": {
      "long": {
        "generic": "שעון ניו זילנד",
        "standard": "שעון ניו זילנד (חורף)",
        "daylight": "שעון ניו זילנד (קיץ)"
      }
    },
    "Newfoundland": {
      "long": {
        "generic": "שעון ניופאונדלנד",
        "standard": "שעון ניופאונדלנד (חורף)",
        "daylight": "שעון ניופאונדלנד (קיץ)"
      }
    },
    "Niue": {
      "long": {
        "standard": "שעון ניואה"
      }
    },
    "Norfolk": {
      "long": {
        "standard": "שעון האי נורפוק"
      }
    },
    "Noronha": {
      "long": {
        "generic": "שעון פרננדו די נורוניה",
        "standard": "שעון פרננדו די נורוניה (חורף)",
        "daylight": "שעון פרננדו די נורוניה (קיץ)"
      }
    },
    "Novosibirsk": {
      "long": {
        "generic": "שעון נובוסיבירסק",
        "standard": "שעון נובוסיבירסק (חורף)",
        "daylight": "שעון נובוסיבירסק (קיץ)"
      }
    },
    "Omsk": {
      "long": {
        "generic": "שעון אומסק",
        "standard": "שעון אומסק (חורף)",
        "daylight": "שעון אומסק (קיץ)"
      }
    },
    "Pakistan": {
      "long": {
        "generic": "שעון פקיסטן",
        "standard": "שעון פקיסטן (חורף)",
        "daylight": "שעון פקיסטן (קיץ)"
      }
    },
    "Palau": {
      "long": {
        "standard": "שעון פלאו"
      }
    }
  }

```

```

    },
    "Papua_New_Guinea": {
      "long": {
        "standard": "שעון פפואה גיניאה החדשה"
      }
    },
    "Paraguay": {
      "long": {
        "generic": "שעון פרגוואי",
        "standard": "שעון פרגוואי (חורף)",
        "daylight": "שעון פרגוואי (קיץ)"
      }
    },
    "Peru": {
      "long": {
        "generic": "שעון פרו",
        "standard": "שעון פרו (חורף)",
        "daylight": "שעון פרו (קיץ)"
      }
    },
    "Philippines": {
      "long": {
        "generic": "שעון הפיליפינים",
        "standard": "שעון הפיליפינים (חורף)",
        "daylight": "שעון הפיליפינים (קיץ)"
      }
    },
    "Phoenix_Islands": {
      "long": {
        "standard": "שעון איי פיניקס"
      }
    },
    "Pierre_Miquelon": {
      "long": {
        "generic": "שעון סנט פייר ומיקלון",
        "standard": "שעון סנט פייר ומיקלון (חורף)",
        "daylight": "שעון סנט פייר ומיקלון (קיץ)"
      }
    },
    "Pitcairn": {
      "long": {
        "standard": "שעון פיטקרן"
      }
    },
    "Ponape": {
      "long": {
        "standard": "שעון פונאפי"
      }
    },
    "Pyongyang": {
      "long": {
        "standard": "שעון פיונגיאנג"
      }
    },
    "Reunion": {
      "long": {
        "standard": "שעון ראוניון"
      }
    }
  }

```



```

    }
  },
  "Rothera": {
    "long": {
      "standard": "שעון רות'רה"
    }
  },
  "Sakhalin": {
    "long": {
      "generic": "שעון סחלין",
      "standard": "שעון סחלין (חורף)",
      "daylight": "שעון סחלין (קיץ)"
    }
  },
  "Samara": {
    "long": {
      "generic": "שעון סמרה",
      "standard": "שעון רגיל סמרה",
      "daylight": "שעון קיץ סמרה"
    }
  },
  "Samoa": {
    "long": {
      "generic": "שעון סמואה",
      "standard": "שעון סמואה (חורף)",
      "daylight": "שעון סמואה (קיץ)"
    }
  },
  "Seychelles": {
    "long": {
      "standard": "שעון איי סיישל"
    }
  },
  "Singapore": {
    "long": {
      "standard": "שעון סינגפור"
    }
  },
  "Solomon": {
    "long": {
      "standard": "שעון איי שלמה"
    }
  },
  "South_Georgia": {
    "long": {
      "standard": "שעון דרום ג'ורג'יה"
    }
  },
  "Suriname": {
    "long": {
      "standard": "שעון סורינאם"
    }
  },
  "Syowa": {
    "long": {
      "standard": "שעון סיווה"
    }
  }
}

```

```

    },
    "Tahiti": {
      "long": {
        "standard": "שעון טהיטי"
      }
    },
    "Taipei": {
      "long": {
        "generic": "שעון טאיפיי",
        "standard": "שעון טאיפיי (חורף)",
        "daylight": "שעון טאיפיי (קיץ)"
      }
    },
    "Tajikistan": {
      "long": {
        "standard": "שעון טג'יקיסטן"
      }
    },
    "Tokelau": {
      "long": {
        "standard": "שעון טוקלאו"
      }
    },
    "Tonga": {
      "long": {
        "generic": "שעון טונגה",
        "standard": "שעון טונגה (חורף)",
        "daylight": "שעון טונגה (קיץ)"
      }
    },
    "Truk": {
      "long": {
        "standard": "שעון צ'וק"
      }
    },
    "Turkmenistan": {
      "long": {
        "generic": "שעון טורקמניסטן",
        "standard": "שעון טורקמניסטן (חורף)",
        "daylight": "שעון טורקמניסטן (קיץ)"
      }
    },
    "Tuvalu": {
      "long": {
        "standard": "שעון טובאלו"
      }
    },
    "Uruguay": {
      "long": {
        "generic": "שעון אורוגוואי",
        "standard": "שעון אורוגוואי (חורף)",
        "daylight": "שעון אורוגוואי (קיץ)"
      }
    },
    "Uzbekistan": {
      "long": {
        "generic": "שעון אוזבקיסטן",

```

```

        "standard": "שעון אוזבקיסטן (חורף)",
        "daylight": "שעון אוזבקיסטן (קיץ)"
    },
},
"Vanuatu": {
    "long": {
        "generic": "שעון ונואטו",
        "standard": "שעון ונואטו (חורף)",
        "daylight": "שעון ונואטו (קיץ)"
    }
},
"Venezuela": {
    "long": {
        "standard": "שעון ונצואלה"
    }
},
"Vladivostok": {
    "long": {
        "generic": "שעון ולדיווסטוק",
        "standard": "שעון ולדיווסטוק (חורף)",
        "daylight": "שעון ולדיווסטוק (קיץ)"
    }
},
"Volgograd": {
    "long": {
        "generic": "שעון וולגוגרד",
        "standard": "שעון וולגוגרד (חורף)",
        "daylight": "שעון וולגוגרד (קיץ)"
    }
},
"Vostok": {
    "long": {
        "standard": "שעון ווסטוק"
    }
},
"Wake": {
    "long": {
        "standard": "שעון האי וייק"
    }
},
"Wallis": {
    "long": {
        "standard": "שעון וואליס ופוטונה"
    }
},
"Yakutsk": {
    "long": {
        "generic": "שעון יקוטסק",
        "standard": "שעון יקוטסק (חורף)",
        "daylight": "שעון יקוטסק (קיץ)"
    }
},
"Yekaterinburg": {
    "long": {
        "generic": "שעון יקטרינבורג",
        "standard": "שעון יקטרינבורג (חורף)",
        "daylight": "שעון יקטרינבורג (קיץ)"
    }
}

```



```
{
  "exemplarCity";
  "אנקורג'";
}
"Anguilla";
{
  "exemplarCity";
  "אנגוויילה";
}
"Antigua";
{
  "exemplarCity";
  "אנטטיגואה";
}
"Araguaina";
{
  "exemplarCity";
  "אראגואינה";
}
"Argentina";
{
  "Rio_Gallegos";
  {
    "exemplarCity";
    "ריו גאליגוס";
  }
  "San_Juan";
  {
    "exemplarCity";
    "סן חואן";
  }
  "Ushuaia";
  {
    "exemplarCity";
    "אושואיה";
  }
  "La_Rioja";
  {
    "exemplarCity";
    "לה ריוחה";
  }
  "San_Luis";
  {
    "exemplarCity";
    "סן לואיס";
  }
  "Salta";
  {
    "exemplarCity";
    "סלטה";
  }
  "Tucuman";
  {
    "exemplarCity";
    "טוקומן";
  }
}
```

```
"Aruba";
{
  "exemplarCity";
  "ארובה";
}
"Asuncion";
{
  "exemplarCity";
  "אסונסיון";
}
"Bahia";
{
  "exemplarCity";
  "באהיה";
}
"Bahia_Banderas";
{
  "exemplarCity";
  "באהיה בנדרס";
}
"Barbados";
{
  "exemplarCity";
  "ברבדוס";
}
"Belem";
{
  "exemplarCity";
  "בלם";
}
"Belize";
{
  "exemplarCity";
  "בליז";
}
"Blanc-Sablon";
{
  "exemplarCity";
  "בלאן-סבלון";
}
"Boa_Vista";
{
  "exemplarCity";
  "בואה ויסטה";
}
"Bogota";
{
  "exemplarCity";
  "בוגוטה";
}
"Boise";
{
  "exemplarCity";
  "בוויסי";
}
"Buenos_Aires";
{
```

```
        "exemplarCity";
        "בואנוס איירס";
    }
    "Cambridge_Bay";
    {
        "exemplarCity";
        "קיימברידג' ביי";
    }
    "Campo_Grande";
    {
        "exemplarCity";
        "קמפו גרנדה";
    }
    "Cancun";
    {
        "exemplarCity";
        "קנקון";
    }
    "Caracas";
    {
        "exemplarCity";
        "קראקס";
    }
    "Catamarca";
    {
        "exemplarCity";
        "קטמרקה";
    }
    "Cayenne";
    {
        "exemplarCity";
        "קאיייל";
    }
    "Cayman";
    {
        "exemplarCity";
        "קיימן";
    }
    "Chicago";
    {
        "exemplarCity";
        "שיקגו";
    }
    "Chihuahua";
    {
        "exemplarCity";
        "צ'יוואוה";
    }
    "Coral_Harbour";
    {
        "exemplarCity";
        "אטיקוקן";
    }
    "Cordoba";
    {
        "exemplarCity";
        "קורדובה";
    }
```

```
}
"Costa_Rica";
{
  "exemplarCity";
  "קוסטה ריקה";
}
"Creston";
{
  "exemplarCity";
  "קרסטון";
}
"Cuiaba";
{
  "exemplarCity";
  "קויאבה";
}
"Curacao";
{
  "exemplarCity";
  "קוראסאו";
}
"Danmarkshavn";
{
  "exemplarCity";
  "דנמרקסהוון";
}
"Dawson";
{
  "exemplarCity";
  "דוסון";
}
"Dawson_Creek";
{
  "exemplarCity";
  "דוסון קריק";
}
"Denver";
{
  "exemplarCity";
  "דנוור";
}
"Detroit";
{
  "exemplarCity";
  "דטרויט";
}
"Dominica";
{
  "exemplarCity";
  "דומיניקה";
}
"Edmonton";
{
  "exemplarCity";
  "אדמונטון";
}
"Eirunepe";
```



```
{
  "exemplarCity";
  "אירונפי";
}
"El_Salvador";
{
  "exemplarCity";
  "אל סלבדור";
}
"Fort_Nelson";
{
  "exemplarCity";
  "פורט נלסון";
}
"Fortaleza";
{
  "exemplarCity";
  "פורטאלזה";
}
"Glace_Bay";
{
  "exemplarCity";
  "גלייס ביי";
}
"Godthab";
{
  "exemplarCity";
  "גואוק";
}
"Goose_Bay";
{
  "exemplarCity";
  "גוס ביי";
}
"Grand_Turk";
{
  "exemplarCity";
  "גרנד טורק";
}
"Grenada";
{
  "exemplarCity";
  "גרנדה";
}
"Guadeloupe";
{
  "exemplarCity";
  "גואדלופ";
}
"Guatemala";
{
  "exemplarCity";
  "גואטמלה";
}
"Guayaquil";
{
  "exemplarCity";
```

```
        "גוֹאִיאַקִּיל";
    }
    "Guyana";
    {
        "exemplarCity";
        "גיאנה";
    }
    "Halifax";
    {
        "exemplarCity";
        "הליפקס";
    }
    "Havana";
    {
        "exemplarCity";
        "הוואנה";
    }
    "Hermosillo";
    {
        "exemplarCity";
        "הרמוסיו";
    }
    "Indiana";
    {
        "Vincennes";
        {
            "exemplarCity";
            "וִינְסֵנְס, אינדיאנה";
        }
        "Petersburg";
        {
            "exemplarCity";
            "פִּיטֶרְסְבוּרְג, אינדיאנה";
        }
        "Tell_City";
        {
            "exemplarCity";
            "טֵל סִיטִי, אינדיאנה";
        }
        "Knox";
        {
            "exemplarCity";
            "נוֹקְס, אינדיאנה";
        }
        "Winamac";
        {
            "exemplarCity";
            "וִינַמַּאק, אינדיאנה";
        }
        "Marengo";
        {
            "exemplarCity";
            "מַרְנֶגו, אינדיאנה";
        }
        "Vevay";
        {
            "exemplarCity";
```

```

        "וירואיי, אינדיאנה";
    }
}
"Indianapolis";
{
    "exemplarCity";
    "אינדיאנאפוליס";
}
"Inuvik";
{
    "exemplarCity";
    "אינוויק";
}
"Iqaluit";
{
    "exemplarCity";
    "איקלואיט";
}
"Jamaica";
{
    "exemplarCity";
    "ג'מייקה";
}
"Jujuy";
{
    "exemplarCity";
    "חוחוי";
}
"Juneau";
{
    "exemplarCity";
    "ג'ונו";
}
"Kentucky";
{
    "Monticello";
    {
        "exemplarCity";
        "מונטיצ'לו, קנטאקי";
    }
}
"Kralendijk";
{
    "exemplarCity";
    "קרלנדייק";
}
"La_Paz";
{
    "exemplarCity";
    "לאה פאז";
}
"Lima";
{
    "exemplarCity";
    "לימה";
}
"Los Angeles";

```

```
{
    "exemplarCity";
    "לוס אנג'לס";
}
"Louisville";
{
    "exemplarCity";
    "לואיוויל";
}
"Lower_Princes";
{
    "exemplarCity";
    "לואוור פרינסס קוורטר";
}
"Maceio";
{
    "exemplarCity";
    "מסיאו";
}
"Managua";
{
    "exemplarCity";
    "מנגואה";
}
"Manaus";
{
    "exemplarCity";
    "מנאוס";
}
"Marigot";
{
    "exemplarCity";
    "מריגו";
}
"Martinique";
{
    "exemplarCity";
    "מרטיניק";
}
"Matamoros";
{
    "exemplarCity";
    "מטמורוס";
}
"Mazatlan";
{
    "exemplarCity";
    "מזטלן";
}
"Mendoza";
{
    "exemplarCity";
    "מנדוזה";
}
"Menominee";
{
    "exemplarCity";
```

```
        "מנומיני";
    }
    "Merida";
    {
        "exemplarCity";
        "מרידה";
    }
    "Metlakatla";
    {
        "exemplarCity";
        "מטלקטלה";
    }
    "Mexico_City";
    {
        "exemplarCity";
        "מקסיקו סיטי";
    }
    "Miquelon";
    {
        "exemplarCity";
        "מיקלון";
    }
    "Moncton";
    {
        "exemplarCity";
        "מונקטון";
    }
    "Monterrey";
    {
        "exemplarCity";
        "מונטריי";
    }
    "Montevideo";
    {
        "exemplarCity";
        "מונטווידאו";
    }
    "Montserrat";
    {
        "exemplarCity";
        "מונטראט";
    }
    "Nassau";
    {
        "exemplarCity";
        "נסאו";
    }
    "New_York";
    {
        "exemplarCity";
        "ניו יורק";
    }
    "Nipigon";
    {
        "exemplarCity";
        "ניפיגון";
    }
}
```

```
"Nome";
{
  "exemplarCity";
  "נום";
}
"Noronha";
{
  "exemplarCity";
  "נורוניה";
}
"North_Dakota";
{
  "Beulah";
  {
    "exemplarCity";
    "ביולה, צפון דקוטה";
  }
  "New_Salem";
  {
    "exemplarCity";
    "ניו סייזם, צפון דקוטה";
  }
  "Center";
  {
    "exemplarCity";
    "סנטר, צפון דקוטה";
  }
}
"Ojinaga";
{
  "exemplarCity";
  "אוג'ינאגה";
}
"Panama";
{
  "exemplarCity";
  "פנמה";
}
"Pangnirtung";
{
  "exemplarCity";
  "פנגנירטונג";
}
"Paramaribo";
{
  "exemplarCity";
  "פרמריבו";
}
"Phoenix";
{
  "exemplarCity";
  "פיניקס";
}
"Port-au-Prince";
{
  "exemplarCity";
  "פורט או פראנס";
}
```

```
}
"Port_of_Spain";
{
  "exemplarCity";
  "פורט אױף ספּײַן";
}
"Porto_Velho";
{
  "exemplarCity";
  "פּױרטױ װליו";
}
"Puerto_Rico";
{
  "exemplarCity";
  "פּױאַרטױ ריקױ";
}
"Rainy_River";
{
  "exemplarCity";
  "רײַני ריבּר";
}
"Rankin_Inlet";
{
  "exemplarCity";
  "רױנקין אינלײט";
}
"Recife";
{
  "exemplarCity";
  "רעסיפּה";
}
"Regina";
{
  "exemplarCity";
  "רגײנאַה";
}
"Resolute";
{
  "exemplarCity";
  "רעזױלוט";
}
"Rio_Branco";
{
  "exemplarCity";
  "ריו בראַנקױ";
}
"Santa_Isabel";
{
  "exemplarCity";
  "סאַנטאַ איזאַבּל";
}
"Santarem";
{
  "exemplarCity";
  "סאַנטאַרעם";
}
"Santiago";
```

```
{
    "exemplarCity";
    "סנט יאגו";
}
"Santo_Domingo";
{
    "exemplarCity";
    "סנטו דומינגו";
}
"Sao_Paulo";
{
    "exemplarCity";
    "סאו פאולו";
}
"Scoresbysund";
{
    "exemplarCity";
    "סקורסביסונד";
}
"Sitka";
{
    "exemplarCity";
    "סיטקה";
}
"St_Barthelemy";
{
    "exemplarCity";
    "סנט ברתלמי";
}
"St_Johns";
{
    "exemplarCity";
    "סנט ג'ונס";
}
"St_Kitts";
{
    "exemplarCity";
    "סנט קיטס";
}
"St_Lucia";
{
    "exemplarCity";
    "סנט לוסיה";
}
"St_Thomas";
{
    "exemplarCity";
    "סנט תומאס";
}
"St_Vincent";
{
    "exemplarCity";
    "סנט וינסנט";
}
"Swift_Current";
{
    "exemplarCity";
```



```
        "סוויפט קרנט";
    }
    "Tegucigalpa";
    {
        "exemplarCity";
        "טגוסיגלפה";
    }
    "Thule";
    {
        "exemplarCity";
        "תולה";
    }
    "Thunder_Bay";
    {
        "exemplarCity";
        "ת'אנדר ביי";
    }
    "Tijuana";
    {
        "exemplarCity";
        "טיחואנה";
    }
    "Toronto";
    {
        "exemplarCity";
        "טורונטו";
    }
    "Tortola";
    {
        "exemplarCity";
        "טורטולה";
    }
    "Vancouver";
    {
        "exemplarCity";
        "ונקובר";
    }
    "Whitehorse";
    {
        "exemplarCity";
        "ווייטהורס";
    }
    "Winnipeg";
    {
        "exemplarCity";
        "וויניפג";
    }
    "Yakutat";
    {
        "exemplarCity";
        "יקוטאט";
    }
    "Yellowknife";
    {
        "exemplarCity";
        "ילוניף";
    }
}
```

```
}
"Atlantic";
{
  "Azores";
  {
    "exemplarCity";
    "האיים האזוריים";
  }
  "Bermuda";
  {
    "exemplarCity";
    "ברמודה";
  }
  "Canary";
  {
    "exemplarCity";
    "האיים הקנריים";
  }
  "Cape_Verde";
  {
    "exemplarCity";
    "כף ורדה";
  }
  "Faeroe";
  {
    "exemplarCity";
    "פארו";
  }
  "Madeira";
  {
    "exemplarCity";
    "מדירה";
  }
  "Reykjavik";
  {
    "exemplarCity";
    "רייקיאויק";
  }
  "South_Georgia";
  {
    "exemplarCity";
    "דרום ג'ורג'יה";
  }
  "St_Helena";
  {
    "exemplarCity";
    "סנט הלנה";
  }
  "Stanley";
  {
    "exemplarCity";
    "סטנלי";
  }
}
"Europe";
{
  "Amsterdam";
```

```
{
  "exemplarCity";
  "אמסטרדם";
}
"Andorra";
{
  "exemplarCity";
  "אנדורה";
}
"Astrakhan";
{
  "exemplarCity";
  "אסטרחן";
}
"Athens";
{
  "exemplarCity";
  "אתונה";
}
"Belgrade";
{
  "exemplarCity";
  "בלגרד";
}
"Berlin";
{
  "exemplarCity";
  "ברלין";
}
"Bratislava";
{
  "exemplarCity";
  "ברטיסלבה";
}
"Brussels";
{
  "exemplarCity";
  "בריסל";
}
"Bucharest";
{
  "exemplarCity";
  "בוקרשט";
}
"Budapest";
{
  "exemplarCity";
  "בודפשט";
}
"Busingen";
{
  "exemplarCity";
  "ביזינגן";
}
"Chisinau";
{
  "exemplarCity";
```

```

        "קייב";
    }
    "Copenhagen";
    {
        "exemplarCity";
        "קופנהגן";
    }
    "Dublin";
    {
        "long";
        {
            "daylight";
            "שעון קיץ אירלנד";
        }
        "exemplarCity";
        "דבלין";
    }
    "Gibraltar";
    {
        "exemplarCity";
        "גיברלטר";
    }
    "Guernsey";
    {
        "exemplarCity";
        "גרנזי";
    }
    "Helsinki";
    {
        "exemplarCity";
        "הלסינקי";
    }
    "Isle_of_Man";
    {
        "exemplarCity";
        "האי מאן";
    }
    "Istanbul";
    {
        "exemplarCity";
        "איסטנבול";
    }
    "Jersey";
    {
        "exemplarCity";
        "ג'רזי";
    }
    "Kaliningrad";
    {
        "exemplarCity";
        "קלינינגרד";
    }
    "Kiev";
    {
        "exemplarCity";
        "קייב";
    }
}

```

```
"Kirov";
{
  "exemplarCity";
  "קִירוֹב";
}
"Lisbon";
{
  "exemplarCity";
  "לִיסְבוֹן";
}
"Ljubljana";
{
  "exemplarCity";
  "לוֹבְלִיאַנָה";
}
"London";
{
  "long";
  {
    "daylight";
    "שְׁעוֹן קִיץ בְּרִיטַנְיָה";
  }
  "exemplarCity";
  "לוֹנְדוֹן";
}
"Luxembourg";
{
  "exemplarCity";
  "לוֹקְסֶמְבוּרְג";
}
"Madrid";
{
  "exemplarCity";
  "מַדְרִיד";
}
"Malta";
{
  "exemplarCity";
  "מַלְטָה";
}
"Mariehamn";
{
  "exemplarCity";
  "מַרִּיהַאֲמֵן";
}
"Minsk";
{
  "exemplarCity";
  "מִינְסְק";
}
"Monaco";
{
  "exemplarCity";
  "מוֹנָקוֹ";
}
"Moscow";
{
```

```
        "exemplarCity";
        "מוסקבה";
    }
    "Oslo";
    {
        "exemplarCity";
        "אוסלו";
    }
    "Paris";
    {
        "exemplarCity";
        "פריז";
    }
    "Podgorica";
    {
        "exemplarCity";
        "פודגוריצה";
    }
    "Prague";
    {
        "exemplarCity";
        "פראג";
    }
    "Riga";
    {
        "exemplarCity";
        "ריגה";
    }
    "Rome";
    {
        "exemplarCity";
        "רומא";
    }
    "Samara";
    {
        "exemplarCity";
        "סמרה";
    }
    "San_Marino";
    {
        "exemplarCity";
        "סן מרינו";
    }
    "Sarajevo";
    {
        "exemplarCity";
        "סרייבו";
    }
    "Simferopol";
    {
        "exemplarCity";
        "סימפרופול";
    }
    "Skopje";
    {
        "exemplarCity";
        "סקופיה";
    }
```

```
}
"Sofia";
{
  "exemplarCity";
  "סופיה";
}
"Stockholm";
{
  "exemplarCity";
  "שטוקהולם";
}
"Tallinn";
{
  "exemplarCity";
  "טאלין";
}
"Tirane";
{
  "exemplarCity";
  "טירנה";
}
"Ulyanovsk";
{
  "exemplarCity";
  "אוליאנובסק";
}
"Uzhgorod";
{
  "exemplarCity";
  "אוז'הורוד";
}
"Vaduz";
{
  "exemplarCity";
  "ואדוץ";
}
"Vatican";
{
  "exemplarCity";
  "הוותיקן";
}
"Vienna";
{
  "exemplarCity";
  "וינה";
}
"Vilnius";
{
  "exemplarCity";
  "וילנה";
}
"Volgograd";
{
  "exemplarCity";
  "וולגוגרד";
}
"Warsaw";
```

```
{
  "exemplarCity";
  "ורשה";
}
"Zagreb";
{
  "exemplarCity";
  "זאגרב";
}
"Zaporozhye";
{
  "exemplarCity";
  "זפורוז'יה";
}
"Zurich";
{
  "exemplarCity";
  "ציריך";
}
}
"Africa";
{
  "Abidjan";
  {
    "exemplarCity";
    "אביג'אן";
  }
  "Accra";
  {
    "exemplarCity";
    "אקרה";
  }
  "Addis_Ababa";
  {
    "exemplarCity";
    "אדיס אבבה";
  }
  "Algiers";
  {
    "exemplarCity";
    "אלג'יר";
  }
  "Asmera";
  {
    "exemplarCity";
    "אסמרה";
  }
  "Bamako";
  {
    "exemplarCity";
    "במאקו";
  }
  "Bangui";
  {
    "exemplarCity";
    "בנגואי";
  }
}
```



```
"Banjul";
{
  "exemplarCity";
  "בנג'ול";
}
"Bissau";
{
  "exemplarCity";
  "ביסאו";
}
"Blantyre";
{
  "exemplarCity";
  "בלנטיר";
}
"Brazzaville";
{
  "exemplarCity";
  "ברזוויל";
}
"Bujumbura";
{
  "exemplarCity";
  "בוג'ומבורה";
}
"Cairo";
{
  "exemplarCity";
  "קהיר";
}
"Casablanca";
{
  "exemplarCity";
  "קזבלנקה";
}
"Ceuta";
{
  "exemplarCity";
  "סאוטה";
}
"Conakry";
{
  "exemplarCity";
  "קונאקרי";
}
"Dakar";
{
  "exemplarCity";
  "דקאר";
}
"Dar_es_Salaam";
{
  "exemplarCity";
  "דאר א-סלאם";
}
"Djibouti";
{
```

```
        "exemplarCity";
        "ג'יבוטי";
    }
    "Douala";
    {
        "exemplarCity";
        "דואלה";
    }
    "El_Aaiun";
    {
        "exemplarCity";
        "אל עיון";
    }
    "Freetown";
    {
        "exemplarCity";
        "פריטאון";
    }
    "Gaborone";
    {
        "exemplarCity";
        "גבורונה";
    }
    "Harare";
    {
        "exemplarCity";
        "הרארה";
    }
    "Johannesburg";
    {
        "exemplarCity";
        "יוהנסבורג";
    }
    "Juba";
    {
        "exemplarCity";
        "ג'ובה";
    }
    "Kampala";
    {
        "exemplarCity";
        "קמפאלה";
    }
    "Khartoum";
    {
        "exemplarCity";
        "חרטום";
    }
    "Kigali";
    {
        "exemplarCity";
        "קיגלי";
    }
    "Kinshasa";
    {
        "exemplarCity";
        "קינשסה";
    }
```

```
}
"Lagos";
{
  "exemplarCity";
  "לגוס";
}
"Libreville";
{
  "exemplarCity";
  "ליברוויל";
}
"Lome";
{
  "exemplarCity";
  "לומה";
}
"Luanda";
{
  "exemplarCity";
  "לואנדה";
}
"Lubumbashi";
{
  "exemplarCity";
  "לובומבאשי";
}
"Lusaka";
{
  "exemplarCity";
  "לוסקה";
}
"Malabo";
{
  "exemplarCity";
  "מלבו";
}
"Maputo";
{
  "exemplarCity";
  "מאפוטו";
}
"Maseru";
{
  "exemplarCity";
  "מסרו";
}
"Mbabane";
{
  "exemplarCity";
  "מבאבאנה";
}
"Mogadishu";
{
  "exemplarCity";
  "מוגדישו";
}
"Monrovia";
```

```

        {
            "exemplarCity";
            "מונרוביה";
        }
        "Nairobi";
        {
            "exemplarCity";
            "ניירובי";
        }
        "Ndjamena";
        {
            "exemplarCity";
            "נג'מנה";
        }
        "Niamey";
        {
            "exemplarCity";
            "ניאמי";
        }
        "Nouakchott";
        {
            "exemplarCity";
            "נואקצ'וט";
        }
        "Ouagadougou";
        {
            "exemplarCity";
            "וואגאדוגו";
        }
        "Porto-Novo";
        {
            "exemplarCity";
            "פורטו נובו";
        }
        "Sao_Tome";
        {
            "exemplarCity";
            "סאו טומה";
        }
        "Tripoli";
        {
            "exemplarCity";
            "טריפולי";
        }
        "Tunis";
        {
            "exemplarCity";
            "תוניס";
        }
        "Windhoek";
        {
            "exemplarCity";
            "ווינדהוק";
        }
    }
    "Asia";
    {

```

```
"Aden";
{
  "exemplarCity";
  "עדן";
}
"Almaty";
{
  "exemplarCity";
  "אלמאטי";
}
"Amman";
{
  "exemplarCity";
  "עמאן";
}
"Anadyr";
{
  "exemplarCity";
  "אנדיר";
}
"Aqtan";
{
  "exemplarCity";
  "אקטאן";
}
"Aqtobe";
{
  "exemplarCity";
  "אקטובה";
}
"Ashgabat";
{
  "exemplarCity";
  "אשגבט";
}
"Baghdad";
{
  "exemplarCity";
  "בגדד";
}
"Bahrain";
{
  "exemplarCity";
  "בחריין";
}
"Baku";
{
  "exemplarCity";
  "באקו";
}
"Bangkok";
{
  "exemplarCity";
  "בנגקוק";
}
"Barnaul";
{
```

```
        "exemplarCity";
        "ברנאול";
    }
    "Beirut";
    {
        "exemplarCity";
        "ביירות";
    }
    "Bishkek";
    {
        "exemplarCity";
        "בישקק";
    }
    "Brunei";
    {
        "exemplarCity";
        "ברוניי";
    }
    "Calcutta";
    {
        "exemplarCity";
        "קולקטה";
    }
    "Chita";
    {
        "exemplarCity";
        "צ'יטה";
    }
    "Choibalsan";
    {
        "exemplarCity";
        "צ'ויבלסן";
    }
    "Colombo";
    {
        "exemplarCity";
        "קולומבו";
    }
    "Damascus";
    {
        "exemplarCity";
        "דמשק";
    }
    "Dhaka";
    {
        "exemplarCity";
        "דאקה";
    }
    "Dili";
    {
        "exemplarCity";
        "דילי";
    }
    "Dubai";
    {
        "exemplarCity";
        "דובאי";
    }
}
```

```
}
"Dushanbe";
{
  "exemplarCity";
  "דושנבה";
}
"Gaza";
{
  "exemplarCity";
  "עזה";
}
"Hebron";
{
  "exemplarCity";
  "חברון";
}
"Hong_Kong";
{
  "exemplarCity";
  "הונג קונג";
}
"Hovd";
{
  "exemplarCity";
  "חובד";
}
"Irkutsk";
{
  "exemplarCity";
  "אירקוטסק";
}
"Jakarta";
{
  "exemplarCity";
  "ג'קרטה";
}
"Jayapura";
{
  "exemplarCity";
  "ג'איאפורה";
}
"Jerusalem";
{
  "exemplarCity";
  "ירושלים";
}
"Kabul";
{
  "exemplarCity";
  "קאבול";
}
"Kamchatka";
{
  "exemplarCity";
  "קמצ'טקה";
}
"Karachi";
```

```
{
    "exemplarCity";
    "קראצ'י";
}
"Katmandu";
{
    "exemplarCity";
    "קטמנדו";
}
"Khandyga";
{
    "exemplarCity";
    "חנדיגה";
}
"Krasnoyarsk";
{
    "exemplarCity";
    "קרסנויארסק";
}
"Kuala_Lumpur";
{
    "exemplarCity";
    "קואלה לומפור";
}
"Kuching";
{
    "exemplarCity";
    "קוצ'ינג";
}
"Kuwait";
{
    "exemplarCity";
    "כווית";
}
"Macau";
{
    "exemplarCity";
    "מקאו";
}
"Magadan";
{
    "exemplarCity";
    "מגדל";
}
"Makassar";
{
    "exemplarCity";
    "מאקאסאר";
}
"Manila";
{
    "exemplarCity";
    "מנילה";
}
"Muscat";
{
    "exemplarCity";
```



```

        "מוסקט";
    }
    "Nicosia";
    {
        "exemplarCity";
        "ניקוסיה";
    }
    "Novokuznetsk";
    {
        "exemplarCity";
        "נובוקוזנטסק";
    }
    "Novosibirsk";
    {
        "exemplarCity";
        "נובוסירסק";
    }
    "Omsk";
    {
        "exemplarCity";
        "אומסק";
    }
    "Oral";
    {
        "exemplarCity";
        "אורל";
    }
    "Phnom_Penh";
    {
        "exemplarCity";
        "פנום פן";
    }
    "Pontianak";
    {
        "exemplarCity";
        "פונטיאנק";
    }
    "Pyongyang";
    {
        "exemplarCity";
        "פיונגיאנג";
    }
    "Qatar";
    {
        "exemplarCity";
        "קטאר";
    }
    "Qyzylorda";
    {
        "exemplarCity";
        "קזילורדה";
    }
    "Rangoon";
    {
        "exemplarCity";
        "רנגון";
    }
}

```

```
"Riyadh";
{
  "exemplarCity";
  "ריאד";
}
"Saigon";
{
  "exemplarCity";
  "הו צ'י מין סיטי";
}
"Sakhalin";
{
  "exemplarCity";
  "סחלין";
}
"Samarkand";
{
  "exemplarCity";
  "סמרקנד";
}
"Seoul";
{
  "exemplarCity";
  "סיאול";
}
"Shanghai";
{
  "exemplarCity";
  "שנחאי";
}
"Singapore";
{
  "exemplarCity";
  "סינגפור";
}
"Srednekolymsk";
{
  "exemplarCity";
  "סרדניקולימסק";
}
"Taipei";
{
  "exemplarCity";
  "טאייפיי";
}
"Tashkent";
{
  "exemplarCity";
  "טשקנט";
}
"Tbilisi";
{
  "exemplarCity";
  "טביליסי";
}
"Tehran";
{
```

```
        "exemplarCity";
        "טהרן";
    }
    "Thimphu";
    {
        "exemplarCity";
        "טהימפהו";
    }
    "Tokyo";
    {
        "exemplarCity";
        "טוקיו";
    }
    "Tomsk";
    {
        "exemplarCity";
        "טומסק";
    }
    "Ulaanbaatar";
    {
        "exemplarCity";
        "אולאאנבטאר";
    }
    "Urumqi";
    {
        "exemplarCity";
        "אורומקי";
    }
    "Ust-Nera";
    {
        "exemplarCity";
        "אוסט-נרה";
    }
    "Vientiane";
    {
        "exemplarCity";
        "האנוי";
    }
    "Vladivostok";
    {
        "exemplarCity";
        "ולדיווסטוק";
    }
    "Yakutsk";
    {
        "exemplarCity";
        "יקוטסק";
    }
    "Yekaterinburg";
    {
        "exemplarCity";
        "יקטרינבורג";
    }
    "Yerevan";
    {
        "exemplarCity";
        "ירוואן";
    }
}
```

```
    }  
  }  
  "Indian";  
  {  
    "Antananarivo";  
    {  
      "exemplarCity";  
      "אנטננריבו";  
    }  
    "Chagos";  
    {  
      "exemplarCity";  
      "צ'אגוס";  
    }  
    "Christmas";  
    {  
      "exemplarCity";  
      "האי כריסטמס";  
    }  
    "Cocos";  
    {  
      "exemplarCity";  
      "קוקוס";  
    }  
    "Comoro";  
    {  
      "exemplarCity";  
      "קומורו";  
    }  
    "Kerguelen";  
    {  
      "exemplarCity";  
      "קרגוולן";  
    }  
    "Mahe";  
    {  
      "exemplarCity";  
      "מהא";  
    }  
    "Maldives";  
    {  
      "exemplarCity";  
      "האיים המלדיביים";  
    }  
    "Mauritius";  
    {  
      "exemplarCity";  
      "מאוריציוס";  
    }  
    "Mayotte";  
    {  
      "exemplarCity";  
      "מאיוט";  
    }  
    "Reunion";  
    {  
      "exemplarCity";  
    }  
  }
```

```
        "ראוניון";
    }
}
"Australia";
{
    "Adelaide";
    {
        "exemplarCity";
        "אדלייד";
    }
    "Brisbane";
    {
        "exemplarCity";
        "בריסביין";
    }
    "Broken_Hill";
    {
        "exemplarCity";
        "ברוקן היל";
    }
    "Currie";
    {
        "exemplarCity";
        "קרי";
    }
    "Darwin";
    {
        "exemplarCity";
        "דרוויין";
    }
    "Eucla";
    {
        "exemplarCity";
        "יוקלה";
    }
    "Hobart";
    {
        "exemplarCity";
        "הוברט";
    }
    "Lindeman";
    {
        "exemplarCity";
        "לינדמן";
    }
    "Lord_Howe";
    {
        "exemplarCity";
        "אי הלורד האו";
    }
    "Melbourne";
    {
        "exemplarCity";
        "מלבורן";
    }
    "Perth";
    {
```

```
        "exemplarCity";
        "פרת'";
    }
    "Sydney";
    {
        "exemplarCity";
        "סידני";
    }
}
"Pacific";
{
    "Apia";
    {
        "exemplarCity";
        "אפיה";
    }
    "Auckland";
    {
        "exemplarCity";
        "אוקלנד";
    }
    "Bougainville";
    {
        "exemplarCity";
        "בוגנוויל";
    }
    "Chatham";
    {
        "exemplarCity";
        "צ'אטהאם";
    }
    "Easter";
    {
        "exemplarCity";
        "איי הפסחא";
    }
    "Efate";
    {
        "exemplarCity";
        "אפטח";
    }
    "Enderbury";
    {
        "exemplarCity";
        "אנדרבורי";
    }
    "Fakaofu";
    {
        "exemplarCity";
        "פקאופו";
    }
    "Fiji";
    {
        "exemplarCity";
        "פיג'י";
    }
    "Funafuti";
```

```
{
  "exemplarCity";
  "פונפוט";
}
"Galapagos";
{
  "exemplarCity";
  "גלפאגוס";
}
"Gambier";
{
  "exemplarCity";
  "אי גמבייה";
}
"Guadalcanal";
{
  "exemplarCity";
  "גוודלקנאל";
}
"Guam";
{
  "exemplarCity";
  "גואם";
}
"Honolulu";
{
  "exemplarCity";
  "הונולולו";
}
"Johnston";
{
  "exemplarCity";
  "ג'ונסטון";
}
"Kiritimati";
{
  "exemplarCity";
  "קיריטימאטי";
}
"Kosrae";
{
  "exemplarCity";
  "קוסרה";
}
"Kwajalein";
{
  "exemplarCity";
  "קוואג'לין";
}
"Majuro";
{
  "exemplarCity";
  "מאג'ורו";
}
"Marquesas";
{
  "exemplarCity";
```

```

        "איי מרקז";
    }
    "Midway";
    {
        "exemplarCity";
        "מידוויי";
    }
    "Nauru";
    {
        "exemplarCity";
        "נאורו";
    }
    "Niue";
    {
        "exemplarCity";
        "ניואה";
    }
    "Norfolk";
    {
        "exemplarCity";
        "נורפוק";
    }
    "Noumea";
    {
        "exemplarCity";
        "נומאה";
    }
    "Pago_Pago";
    {
        "exemplarCity";
        "פאגו פאגו";
    }
    "Palau";
    {
        "exemplarCity";
        "פלאו";
    }
    "Pitcairn";
    {
        "exemplarCity";
        "פיטקרן";
    }
    "Ponape";
    {
        "exemplarCity";
        "פונפיי";
    }
    "Port_Moresby";
    {
        "exemplarCity";
        "פורט מורסבי";
    }
    "Rarotonga";
    {
        "exemplarCity";
        "רארוטונגה";
    }
}

```



```
"Saipan";
{
  "exemplarCity";
  "סאַיפאַן";
}
"Tahiti";
{
  "exemplarCity";
  "טהיטי";
}
"Tarawa";
{
  "exemplarCity";
  "טאַראווה";
}
"Tongatapu";
{
  "exemplarCity";
  "טונגטאַפּוּ";
}
"Truk";
{
  "exemplarCity";
  "צ'וק";
}
"Wake";
{
  "exemplarCity";
  "ווייק";
}
"Wallis";
{
  "exemplarCity";
  "וואליס";
}
}
"Arctic";
{
  "Longyearbyen";
  {
    "exemplarCity";
    "לונגייַרבײַן";
  }
}
"Antarctica";
{
  "Casey";
  {
    "exemplarCity";
    "קאַסיי";
  }
  "Davis";
  {
    "exemplarCity";
    "דיוויס";
  }
  "DumontDUrville";
```

```

    {
      "exemplarCity";
      "דומון דיאורוויל";
    }
    "Macquarie";
    {
      "exemplarCity";
      "מקרי";
    }
    "Mawson";
    {
      "exemplarCity";
      "מוסון";
    }
    "McMurdo";
    {
      "exemplarCity";
      "מק-מרדו";
    }
    "Palmer";
    {
      "exemplarCity";
      "פאלמר";
    }
    "Rothera";
    {
      "exemplarCity";
      "רות'רה";
    }
    "Syowa";
    {
      "exemplarCity";
      "סיוואה";
    }
    "Troll";
    {
      "exemplarCity";
      "טרול";
    }
    "Vostok";
    {
      "exemplarCity";
      "ווסטוק";
    }
  }
  "Etc";
  {
    "GMT";
    {
      "exemplarCity";
      "GMT";
    }
    "GMT1";
    {
      "exemplarCity";
      "GMT+1";
    }
  }

```

```
"GMT10";
{
  "exemplarCity";
  "GMT+10";
}
"GMT11";
{
  "exemplarCity";
  "GMT+11";
}
"GMT12";
{
  "exemplarCity";
  "GMT+12";
}
"GMT2";
{
  "exemplarCity";
  "GMT+2";
}
"GMT3";
{
  "exemplarCity";
  "GMT+3";
}
"GMT4";
{
  "exemplarCity";
  "GMT+4";
}
"GMT5";
{
  "exemplarCity";
  "GMT+5";
}
"GMT6";
{
  "exemplarCity";
  "GMT+6";
}
"GMT7";
{
  "exemplarCity";
  "GMT+7";
}
"GMT8";
{
  "exemplarCity";
  "GMT+8";
}
"GMT9";
{
  "exemplarCity";
  "GMT+9";
}
"GMT-1";
{
```

```
        "exemplarCity";
        "GMT-1";
    }
    "GMT-10";
    {
        "exemplarCity";
        "GMT-10";
    }
    "GMT-11";
    {
        "exemplarCity";
        "GMT-11";
    }
    "GMT-12";
    {
        "exemplarCity";
        "GMT-12";
    }
    "GMT-13";
    {
        "exemplarCity";
        "GMT-13";
    }
    "GMT-14";
    {
        "exemplarCity";
        "GMT-14";
    }
    "GMT-2";
    {
        "exemplarCity";
        "GMT-2";
    }
    "GMT-3";
    {
        "exemplarCity";
        "GMT-3";
    }
    "GMT-4";
    {
        "exemplarCity";
        "GMT-4";
    }
    "GMT-5";
    {
        "exemplarCity";
        "GMT-5";
    }
    "GMT-6";
    {
        "exemplarCity";
        "GMT-6";
    }
    "GMT-7";
    {
        "exemplarCity";
        "GMT-7";
    }
```

```

    }
    "GMT-8";
    {
      "exemplarCity";
      "GMT-8";
    }
    "GMT-9";
    {
      "exemplarCity";
      "GMT-9";
    }
    "UTC";
    {
      "long";
      {
        "standard";
        "זמן אוניברסלי מתואם";
      }
      "short";
      {
        "standard";
        "UTC";
      }
      "exemplarCity";
      "UTC";
    }
    "Unknown";
    {
      "exemplarCity";
      "עיר לא ידועה";
    }
  }
}
"metazone";
{
  "Afghanistan";
  {
    "long";
    {
      "standard";
      "שעון אפגניסטן";
    }
  }
  "Africa_Central";
  {
    "long";
    {
      "standard";
      "שעון מרכז אפריקה";
    }
  }
  "Africa_Eastern";
  {
    "long";
    {
      "standard";
      "שעון מזרח אפריקה";
    }
  }
}

```

```
    }  
  }  
  "Africa_Southern";  
  {  
    "long";  
    {  
      "standard";  
      "שעון דרום אפריקה";  
    }  
  }  
  "Africa_Western";  
  {  
    "long";  
    {  
      "generic";  
      "שעון מערב אפריקה",  
      "standard";  
      "שעון מערב אפריקה (חורף)",  
      "daylight";  
      "שעון מערב אפריקה (קיץ)";  
    }  
  }  
  "Alaska";  
  {  
    "long";  
    {  
      "generic";  
      "שעון אלסקה",  
      "standard";  
      "שעון אלסקה (חורף)",  
      "daylight";  
      "שעון אלסקה (קיץ)";  
    }  
  }  
  "Amazon";  
  {  
    "long";  
    {  
      "generic";  
      "שעון אמזונס",  
      "standard";  
      "שעון אמזונס (חורף)",  
      "daylight";  
      "שעון אמזונס (קיץ)";  
    }  
  }  
  "America_Central";  
  {  
    "long";  
    {  
      "generic";  
      "שעון מרכז ארה"ב",  
      "standard";  
      "שעון מרכז ארה"ב (חורף)",  
      "daylight";  
      "שעון מרכז ארה"ב (קיץ)";  
    }  
  }
```

```
}
"America_Eastern";
{
  "long";
  {
    "generic";
    "שעון החוף המזרחי",
    "standard";
    "שעון החוף המזרחי (חורף)",
    "daylight";
    "שעון החוף המזרחי (קיץ)";
  }
}
"America_Mountain";
{
  "long";
  {
    "generic";
    "שעון אזור ההרים בארה"ב",
    "standard";
    "שעון אזור ההרים בארה"ב (חורף)",
    "daylight";
    "שעון אזור ההרים בארה"ב (קיץ)";
  }
}
"America_Pacific";
{
  "long";
  {
    "generic";
    "שעון מערב ארה"ב",
    "standard";
    "שעון מערב ארה"ב (חורף)",
    "daylight";
    "שעון מערב ארה"ב (קיץ)";
  }
}
"Anadyr";
{
  "long";
  {
    "generic";
    "שעון אנדיר",
    "standard";
    "שעון רגיל אנדיר",
    "daylight";
    "שעון קיץ אנדיר";
  }
}
"Apia";
{
  "long";
  {
    "generic";
    "שעון אפיה",
    "standard";
    "שעון אפיה (חורף)",
```

```

        "daylight";
        "שעון אפיה (קיץ)";
    }
}
"Arabian";
{
    "long";
    {
        "generic";
        "שעון חצי האי ערב",
        "standard";
        "שעון חצי האי ערב (חורף)",
        "daylight";
        "שעון חצי האי ערב (קיץ)";
    }
}
"Argentina";
{
    "long";
    {
        "generic";
        "שעון ארגנטינה",
        "standard";
        "שעון ארגנטינה (חורף)",
        "daylight";
        "שעון ארגנטינה (קיץ)";
    }
}
"Argentina_Western";
{
    "long";
    {
        "generic";
        "שעון מערב ארגנטינה",
        "standard";
        "שעון מערב ארגנטינה (חורף)",
        "daylight";
        "שעון מערב ארגנטינה (קיץ)";
    }
}
"Armenia";
{
    "long";
    {
        "generic";
        "שעון ארמניה",
        "standard";
        "שעון ארמניה (חורף)",
        "daylight";
        "שעון ארמניה (קיץ)";
    }
}
"Atlantic";
{
    "long";
    {
        "generic";

```



```

        "שעון האוקיינוס האטלנטי",
        "standard";
        "שעון האוקיינוס האטלנטי (חורף)",
        "daylight";
        "שעון האוקיינוס האטלנטי (קיץ)";
    }
}
"Australia_Central";
{
    "long";
    {
        "generic";
        "שעון מרכז אוסטרליה",
        "standard";
        "שעון מרכז אוסטרליה (חורף)",
        "daylight";
        "שעון מרכז אוסטרליה (קיץ)";
    }
}
"Australia_CentralWestern";
{
    "long";
    {
        "generic";
        "שעון מרכז-מערב אוסטרליה",
        "standard";
        "שעון מרכז-מערב אוסטרליה (חורף)",
        "daylight";
        "שעון מרכז-מערב אוסטרליה (קיץ)";
    }
}
"Australia_Eastern";
{
    "long";
    {
        "generic";
        "שעון מזרח אוסטרליה",
        "standard";
        "שעון מזרח אוסטרליה (חורף)",
        "daylight";
        "שעון מזרח אוסטרליה (קיץ)";
    }
}
"Australia_Western";
{
    "long";
    {
        "generic";
        "שעון מערב אוסטרליה",
        "standard";
        "שעון מערב אוסטרליה (חורף)",
        "daylight";
        "שעון מערב אוסטרליה (קיץ)";
    }
}
"Azerbaijan";
{

```

```

        "long";
        {
            "generic";
            "שעון אזרבייג'אן",
            "standard";
            "שעון אזרבייג'אן (חורף)",
            "daylight";
            "שעון אזרבייג'אן (קיץ)";
        }
    }
    "Azores";
    {
        "long";
        {
            "generic";
            "שעון האיים האזוריים",
            "standard";
            "שעון האיים האזוריים (חורף)",
            "daylight";
            "שעון האיים האזוריים (קיץ)";
        }
    }
    "Bangladesh";
    {
        "long";
        {
            "generic";
            "שעון בנגלדש",
            "standard";
            "שעון בנגלדש (חורף)",
            "daylight";
            "שעון בנגלדש (קיץ)";
        }
    }
    "Bhutan";
    {
        "long";
        {
            "standard";
            "שעון בהוטן";
        }
    }
    "Bolivia";
    {
        "long";
        {
            "standard";
            "שעון בוליביה";
        }
    }
    "Brasilia";
    {
        "long";
        {
            "generic";
            "שעון ברזיליה",
            "standard";

```

```
        "שעון ברזיליה (חורף)",
        "daylight";
        "שעון ברזיליה (קיץ)";
    }
}
"Brunei";
{
    "long";
    {
        "standard";
        "שעון ברוניי דארוסלאם";
    }
}
"Cape_Verde";
{
    "long";
    {
        "generic";
        "שעון כף ורדה",
        "standard";
        "שעון כף ורדה (חורף)",
        "daylight";
        "שעון כף ורדה (קיץ)";
    }
}
"Chamorro";
{
    "long";
    {
        "standard";
        "שעון צ'אמורו";
    }
}
"Chatham";
{
    "long";
    {
        "generic";
        "שעון צ'טהאם",
        "standard";
        "שעון צ'טהאם (חורף)",
        "daylight";
        "שעון צ'טהאם (קיץ)";
    }
}
"Chile";
{
    "long";
    {
        "generic";
        "שעון צ'ילה",
        "standard";
        "שעון צ'ילה (חורף)",
        "daylight";
        "שעון צ'ילה (קיץ)";
    }
}
}
```

```
"China";
{
  "long";
  {
    "generic";
    "שעון סין",
    "standard";
    "שעון סין (חורף)",
    "daylight";
    "שעון סין (קיץ)";
  }
}
"Choibalsan";
{
  "long";
  {
    "generic";
    "שעון צ'ויבלסן",
    "standard";
    "שעון צ'ויבלסן (חורף)",
    "daylight";
    "שעון צ'ויבלסן (קיץ)";
  }
}
"Christmas";
{
  "long";
  {
    "standard";
    "שעון האי כריסטמס";
  }
}
"Cocos";
{
  "long";
  {
    "standard";
    "שעון איי קוקוס";
  }
}
"Colombia";
{
  "long";
  {
    "generic";
    "שעון קולומביה",
    "standard";
    "שעון קולומביה (חורף)",
    "daylight";
    "שעון קולומביה (קיץ)";
  }
}
"Cook";
{
  "long";
  {
    "generic";
```

```

        "שעון איי קוק",
        "standard";
        "שעון איי קוק (חורף)",
        "daylight";
        "שעון איי קוק (מחצית הקיץ)";
    }
}
"Cuba";
{
    "long";
    {
        "generic";
        "שעון קובה",
        "standard";
        "שעון קובה (חורף)",
        "daylight";
        "שעון קובה (קיץ)";
    }
}
"Davis";
{
    "long";
    {
        "standard";
        "שעון דייוויס";
    }
}
"DumontDUrville";
{
    "long";
    {
        "standard";
        "שעון דומון ד'אורוויל";
    }
}
"East_Timor";
{
    "long";
    {
        "standard";
        "שעון מזרח טימור";
    }
}
"Easter";
{
    "long";
    {
        "generic";
        "שעון אי הפסחא",
        "standard";
        "שעון אי הפסחא (חורף)",
        "daylight";
        "שעון אי הפסחא (קיץ)";
    }
}
"Ecuador";
{

```

```
        "long";
        {
            "standard";
            "שעון אקוודור";
        }
    }
    "Europe_Central";
    {
        "long";
        {
            "generic";
            "שעון מרכז אירופה",
            "standard";
            "שעון מרכז אירופה (חורף)",
            "daylight";
            "שעון מרכז אירופה (קיץ)";
        }
    }
    "Europe_Eastern";
    {
        "long";
        {
            "generic";
            "שעון מזרח אירופה",
            "standard";
            "שעון מזרח אירופה (חורף)",
            "daylight";
            "שעון מזרח אירופה (קיץ)";
        }
    }
    "Europe_Further_Eastern";
    {
        "long";
        {
            "standard";
            "שעון מינסק";
        }
    }
    "Europe_Western";
    {
        "long";
        {
            "generic";
            "שעון מערב אירופה",
            "standard";
            "שעון מערב אירופה (חורף)",
            "daylight";
            "שעון מערב אירופה (קיץ)";
        }
    }
    "Falkland";
    {
        "long";
        {
            "generic";
            "שעון איי פוקלנד",
            "standard";
        }
    }
}
```

```

        "שעון איי פוקלנד (חורף)",
        "daylight";
        "שעון איי פוקלנד (קיץ)";
    }
}
"Fiji";
{
    "long";
    {
        "generic";
        "שעון פיג'י",
        "standard";
        "שעון פיג'י (חורף)",
        "daylight";
        "שעון פיג'י (קיץ)";
    }
}
"French_Guiana";
{
    "long";
    {
        "standard";
        "שעון גיאנה הצרפתית";
    }
}
"French_Southern";
{
    "long";
    {
        "standard";
        "שעון הארצות הדרומיות והאנטארקטיות של צרפת";
    }
}
"Galapagos";
{
    "long";
    {
        "standard";
        "שעון איי גלאפגוס";
    }
}
"Gambier";
{
    "long";
    {
        "standard";
        "שעון איי גמבייה";
    }
}
"Georgia";
{
    "long";
    {
        "generic";
        "שעון גאורגיה",
        "standard";
        "שעון גאורגיה (חורף)",

```

```

        "daylight";
        "שעון גאורגיה (קיץ)";
    }
}
"Gilbert_Islands";
{
    "long";
    {
        "standard";
        "שעון איי גילברט";
    }
}
"GMT";
{
    "long";
    {
        "standard";
        "שעון גריניץ'";
    }
}
"Greenland_Eastern";
{
    "long";
    {
        "generic";
        "שעון מזרח גרינלנד",
        "standard";
        "שעון מזרח גרינלנד (חורף)",
        "daylight";
        "שעון מזרח גרינלנד (קיץ)";
    }
}
"Greenland_Western";
{
    "long";
    {
        "generic";
        "שעון מערב גרינלנד",
        "standard";
        "שעון מערב גרינלנד (חורף)",
        "daylight";
        "שעון מערב גרינלנד (קיץ)";
    }
}
"Gulf";
{
    "long";
    {
        "standard";
        "שעון מדינות המפרץ";
    }
}
"Guyana";
{
    "long";
    {
        "standard";
    }
}

```



```
        "שעון גיאנה";
    }
}
"Hawaii_Aleutian";
{
    "long";
    {
        "generic";
        "שעון האיים האלאוטיים הוואי",
        "standard";
        "שעון האיים האלאוטיים הוואי (חורף)",
        "daylight";
        "שעון האיים האלאוטיים הוואי (קיץ)";
    }
}
"Hong_Kong";
{
    "long";
    {
        "generic";
        "שעון הונג קונג",
        "standard";
        "שעון הונג קונג (חורף)",
        "daylight";
        "שעון הונג קונג (קיץ)";
    }
}
"Hovd";
{
    "long";
    {
        "generic";
        "שעון חובד",
        "standard";
        "שעון חובד (חורף)",
        "daylight";
        "שעון חובד (קיץ)";
    }
}
"India";
{
    "long";
    {
        "standard";
        "שעון הודו";
    }
}
"Indian_Ocean";
{
    "long";
    {
        "standard";
        "שעון האוקיינוס ההודי";
    }
}
"Indochina";
{
```

```
        "long";
        {
            "standard";
            "שעון הודו-סין";
        }
    }
    "Indonesia_Central";
    {
        "long";
        {
            "standard";
            "שעון מרכז אינדונזיה";
        }
    }
    "Indonesia_Eastern";
    {
        "long";
        {
            "standard";
            "שעון מזרח אינדונזיה";
        }
    }
    "Indonesia_Western";
    {
        "long";
        {
            "standard";
            "שעון מערב אינדונזיה";
        }
    }
    "Iran";
    {
        "long";
        {
            "generic";
            "שעון איראן",
            "standard";
            "שעון איראן (חורף)",
            "daylight";
            "שעון איראן (קיץ)";
        }
    }
    "Irkutsk";
    {
        "long";
        {
            "generic";
            "שעון אירקוטסק",
            "standard";
            "שעון אירקוטסק (חורף)",
            "daylight";
            "שעון אירקוטסק (קיץ)";
        }
    }
    "Israel";
    {
        "long";
```

```

        {
            "generic";
            "שעון ישראל",
            "standard";
            "שעון ישראל (חורף)",
            "daylight";
            "שעון ישראל (קיץ)";
        }
    }
    "Japan";
    {
        "long";
        {
            "generic";
            "שעון יפן",
            "standard";
            "שעון יפן (חורף)",
            "daylight";
            "שעון יפן (קיץ)";
        }
    }
    "Kamchatka";
    {
        "long";
        {
            "generic";
            "שעון פטרופלובסק-קמצ'טסקי",
            "standard";
            "שעון רגיל פטרופלובסק-קמצ'טסקי",
            "daylight";
            "שעון קיץ פטרופלובסק-קמצ'טסקי";
        }
    }
    "Kazakhstan_Eastern";
    {
        "long";
        {
            "standard";
            "שעון מזרח קזחסטן";
        }
    }
    "Kazakhstan_Western";
    {
        "long";
        {
            "standard";
            "שעון מערב קזחסטן";
        }
    }
    "Korea";
    {
        "long";
        {
            "generic";
            "שעון קוריאה",
            "standard";
            "שעון קוריאה (חורף)",

```

```

        "daylight";
        "שעון קוריאה (קיץ)";
    }
}
"Kosrae";
{
    "long";
    {
        "standard";
        "שעון קוסראה";
    }
}
"Krasnoyarsk";
{
    "long";
    {
        "generic";
        "שעון קרסנויארסק",
        "standard";
        "שעון קרסנויארסק (חורף)",
        "daylight";
        "שעון קרסנויארסק (קיץ)";
    }
}
"Kyrgystan";
{
    "long";
    {
        "standard";
        "שעון קירגיזסטן";
    }
}
"Line_Islands";
{
    "long";
    {
        "standard";
        "שעון איי ליין";
    }
}
"Lord_Howe";
{
    "long";
    {
        "generic";
        "שעון אי הלורד האו",
        "standard";
        "שעון אי הלורד האו (חורף)",
        "daylight";
        "שעון אי הלורד האו (קיץ)";
    }
}
"Macau";
{
    "long";
    {
        "generic";
    }
}

```

```

        "שעון מקאו",
        "standard";
        "שעון חורף מקאו",
        "daylight";
        "שעון קיץ מקאו";
    }
}
"Macquarie";
{
    "long";
    {
        "standard";
        "שעון מקווארי";
    }
}
"Magadan";
{
    "long";
    {
        "generic";
        "שעון מגדן",
        "standard";
        "שעון מגדן (חורף)",
        "daylight";
        "שעון מגדן (קיץ)";
    }
}
"Malaysia";
{
    "long";
    {
        "standard";
        "שעון מלזיה";
    }
}
"Maldives";
{
    "long";
    {
        "standard";
        "שעון האיים המלדיביים";
    }
}
"Marquesas";
{
    "long";
    {
        "standard";
        "שעון איי מרקז";
    }
}
"Marshall_Islands";
{
    "long";
    {
        "standard";
        "שעון איי מרשל";
    }
}

```

```
    }  
  }  
  "Mauritius";  
  {  
    "long";  
    {  
      "generic";  
      "שעון מאוריציוס",  
      "standard";  
      "שעון מאוריציוס (חורף)",  
      "daylight";  
      "שעון מאוריציוס (קיץ)";  
    }  
  }  
  "Mawson";  
  {  
    "long";  
    {  
      "standard";  
      "שעון מאוסון";  
    }  
  }  
  "Mexico_Northwest";  
  {  
    "long";  
    {  
      "generic";  
      "שעון צפון-מערב מקסיקו",  
      "standard";  
      "שעון צפון-מערב מקסיקו (חורף)",  
      "daylight";  
      "שעון צפון-מערב מקסיקו (קיץ)";  
    }  
  }  
  "Mexico_Pacific";  
  {  
    "long";  
    {  
      "generic";  
      "שעון מערב מקסיקו",  
      "standard";  
      "שעון מערב מקסיקו (חורף)",  
      "daylight";  
      "שעון מערב מקסיקו (קיץ)";  
    }  
  }  
  "Mongolia";  
  {  
    "long";  
    {  
      "generic";  
      "שעון אולן בטור",  
      "standard";  
      "שעון אולן בטור (חורף)",  
      "daylight";  
      "שעון אולן בטור (קיץ)";  
    }  
  }  
}
```

```
}
"Moscow";
{
  "long";
  {
    "generic";
    "שעון מוסקבה",
    "standard";
    "שעון מוסקבה (חורף)",
    "daylight";
    "שעון מוסקבה (קיץ)";
  }
}
"Myanmar";
{
  "long";
  {
    "standard";
    "שעון מיאנמר";
  }
}
"Nauru";
{
  "long";
  {
    "standard";
    "שעון נאורו";
  }
}
"Nepal";
{
  "long";
  {
    "standard";
    "שעון נפאל";
  }
}
"New_Caledonia";
{
  "long";
  {
    "generic";
    "שעון קלדוניה החדשה",
    "standard";
    "שעון קלדוניה החדשה (חורף)",
    "daylight";
    "שעון קלדוניה החדשה (קיץ)";
  }
}
"New_Zealand";
{
  "long";
  {
    "generic";
    "שעון ניו זילנד",
    "standard";
    "שעון ניו זילנד (חורף)",

```

```
        "daylight";
        "שעון ניו זילנד (קיץ)";
    }
}
"Newfoundland";
{
    "long";
    {
        "generic";
        "שעון ניופאונדלנד",
        "standard";
        "שעון ניופאונדלנד (חורף)",
        "daylight";
        "שעון ניופאונדלנד (קיץ)";
    }
}
"Niue";
{
    "long";
    {
        "standard";
        "שעון ניואה";
    }
}
"Norfolk";
{
    "long";
    {
        "standard";
        "שעון האי נורפוק";
    }
}
"Noronha";
{
    "long";
    {
        "generic";
        "שעון פרננדו די נורוניה",
        "standard";
        "שעון פרננדו די נורוניה (חורף)",
        "daylight";
        "שעון פרננדו די נורוניה (קיץ)";
    }
}
"Novosibirsk";
{
    "long";
    {
        "generic";
        "שעון נובוסיבירסק",
        "standard";
        "שעון נובוסיבירסק (חורף)",
        "daylight";
        "שעון נובוסיבירסק (קיץ)";
    }
}
"Omsk";
```



```

    {
      "long";
      {
        "generic";
        "שעון אומסק",
        "standard";
        "שעון אומסק (חורף)",
        "daylight";
        "שעון אומסק (קיץ)";
      }
    }
    "Pakistan";
    {
      "long";
      {
        "generic";
        "שעון פקיסטן",
        "standard";
        "שעון פקיסטן (חורף)",
        "daylight";
        "שעון פקיסטן (קיץ)";
      }
    }
    "Palau";
    {
      "long";
      {
        "standard";
        "שעון פלאו";
      }
    }
    "Papua_New_Guinea";
    {
      "long";
      {
        "standard";
        "שעון פפואה גיניאה החדשה";
      }
    }
    "Paraguay";
    {
      "long";
      {
        "generic";
        "שעון פרגוואי",
        "standard";
        "שעון פרגוואי (חורף)",
        "daylight";
        "שעון פרגוואי (קיץ)";
      }
    }
    "Peru";
    {
      "long";
      {
        "generic";
        "שעון פרו",

```

```

        "standard";
        "שעון פרז (חורף)",
        "daylight";
        "שעון פרז (קיץ)";
    }
}
"Philippines";
{
    "long";
    {
        "generic";
        "שעון הפיליפינים",
        "standard";
        "שעון הפיליפינים (חורף)",
        "daylight";
        "שעון הפיליפינים (קיץ)";
    }
}
"Phoenix_Islands";
{
    "long";
    {
        "standard";
        "שעון איי פניקס";
    }
}
"Pierre_Miquelon";
{
    "long";
    {
        "generic";
        "שעון סנט פייר ומיקלון",
        "standard";
        "שעון סנט פייר ומיקלון (חורף)",
        "daylight";
        "שעון סנט פייר ומיקלון (קיץ)";
    }
}
"Pitcairn";
{
    "long";
    {
        "standard";
        "שעון פיטקרן";
    }
}
"Ponape";
{
    "long";
    {
        "standard";
        "שעון פונאפי";
    }
}
"Pyongyang";
{
    "long";

```

```

        {
            "standard";
            "שעון פיונגיאנג";
        }
    }
    "Reunion";
    {
        "long";
        {
            "standard";
            "שעון ראוניון";
        }
    }
    "Rothera";
    {
        "long";
        {
            "standard";
            "שעון רות'רה";
        }
    }
    "Sakhalin";
    {
        "long";
        {
            "generic";
            "שעון סחלין",
            "standard";
            "שעון סחלין (חורף)",
            "daylight";
            "שעון סחלין (קיץ)";
        }
    }
    "Samara";
    {
        "long";
        {
            "generic";
            "שעון סמרה",
            "standard";
            "שעון רגיל סמרה",
            "daylight";
            "שעון קיץ סמרה";
        }
    }
    "Samoa";
    {
        "long";
        {
            "generic";
            "שעון סמואה",
            "standard";
            "שעון סמואה (חורף)",
            "daylight";
            "שעון סמואה (קיץ)";
        }
    }
}

```

```
"Seychelles";
{
  "long";
  {
    "standard";
    "שעון איי סיישל";
  }
}
"Singapore";
{
  "long";
  {
    "standard";
    "שעון סינגפור";
  }
}
"Solomon";
{
  "long";
  {
    "standard";
    "שעון איי שלמה";
  }
}
"South_Georgia";
{
  "long";
  {
    "standard";
    "שעון דרום ג'ורג'יה";
  }
}
"Suriname";
{
  "long";
  {
    "standard";
    "שעון סורינאם";
  }
}
"Syowa";
{
  "long";
  {
    "standard";
    "שעון סייווה";
  }
}
"Tahiti";
{
  "long";
  {
    "standard";
    "שעון טהיטי";
  }
}
"Taipei";
```

```

    {
      "long";
      {
        "generic";
        "שעון טאיפיי",
        "standard";
        "שעון טאיפיי (חורף)",
        "daylight";
        "שעון טאיפיי (קיץ)";
      }
    }
    "Tajikistan";
    {
      "long";
      {
        "standard";
        "שעון טג'יקיסטן";
      }
    }
    "Tokelau";
    {
      "long";
      {
        "standard";
        "שעון טוקלאו";
      }
    }
    "Tonga";
    {
      "long";
      {
        "generic";
        "שעון טונגה",
        "standard";
        "שעון טונגה (חורף)",
        "daylight";
        "שעון טונגה (קיץ)";
      }
    }
    "Truk";
    {
      "long";
      {
        "standard";
        "שעון צ'וק";
      }
    }
    "Turkmenistan";
    {
      "long";
      {
        "generic";
        "שעון טורקמניסטן",
        "standard";
        "שעון טורקמניסטן (חורף)",
        "daylight";
        "שעון טורקמניסטן (קיץ)";
      }
    }

```

```

    }
  }
  "Tuvalu";
  {
    "long";
    {
      "standard";
      "שעון טובאלו";
    }
  }
  "Uruguay";
  {
    "long";
    {
      "generic";
      "שעון אורוגוואי",
      "standard";
      "שעון אורוגוואי (חורף)",
      "daylight";
      "שעון אורוגוואי (קיץ)";
    }
  }
  "Uzbekistan";
  {
    "long";
    {
      "generic";
      "שעון אוזבקיסטן",
      "standard";
      "שעון אוזבקיסטן (חורף)",
      "daylight";
      "שעון אוזבקיסטן (קיץ)";
    }
  }
  "Vanuatu";
  {
    "long";
    {
      "generic";
      "שעון ונואטו",
      "standard";
      "שעון ונואטו (חורף)",
      "daylight";
      "שעון ונואטו (קיץ)";
    }
  }
  "Venezuela";
  {
    "long";
    {
      "standard";
      "שעון ונצואלה";
    }
  }
  "Vladivostok";
  {
    "long";
  }

```

```

        {
            "generic";
            "שעון ולדיווסטוק",
            "standard";
            "שעון ולדיווסטוק (חורף)",
            "daylight";
            "שעון ולדיווסטוק (קיץ)";
        }
    }
    "Volgograd";
    {
        "long";
        {
            "generic";
            "שעון וולגוגרד",
            "standard";
            "שעון וולגוגרד (חורף)",
            "daylight";
            "שעון וולגוגרד (קיץ)";
        }
    }
    "Vostok";
    {
        "long";
        {
            "standard";
            "שעון ווסטוק";
        }
    }
    "Wake";
    {
        "long";
        {
            "standard";
            "שעון האי וייק";
        }
    }
    "Wallis";
    {
        "long";
        {
            "standard";
            "שעון וואליס ופוטונה";
        }
    }
    "Yakutsk";
    {
        "long";
        {
            "generic";
            "שעון יקוטסק",
            "standard";
            "שעון יקוטסק (חורף)",
            "daylight";
            "שעון יקוטסק (קיץ)";
        }
    }
}

```

```
"Yekaterinburg";
{
    "long";
    {
        "generic";
        "שעון יקטרינבורג",
        "standard";
        "שעון יקטרינבורג (חורף)",
        "daylight";
        "שעון יקטרינבורג (קיץ)";
    }
}
}
```

Strict mode in React Datepicker component

The [strictMode](#) is an act, that allows the user to enter only the valid date within the specified min/max range in textbox. If the date is invalid, then the component will stay with the previous value.

Else, if the date is out of range, then the component will set the date to the min/max date.

The following example demonstrates the DatePicker in `strictMode` with min/max range of 5th to 25th in a month of May. Here, it allows to enter only the valid date within the specified range. If you are trying to enter the out-of-range value as like 28th of May, then the value will set to the max date of 25th May. Since the value 28th is greater than to `max` value of 25th. Or else if you are trying to enter the invalid date, then the value will stay with the previous value.

[Class-component]

INDEX.JSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';

// creates a datepicker with strictMode property
export default class App extends React.Component {
  // sets the value
  dateValue = new Date('5/28/2017');
  // sets the min
  minDate = new Date('5/5/2017');
  // sets the max
  maxDate = new Date('5/25/2017');
  disable = true;

  render() {
    return <DatePickerComponent id="datepicker" strictMode={this.disable}
    format="dd/MM/yyyy" value={this.dateValue} min={this.minDate}
    max={this.maxDate} placeholder="Enter date"/>;
  }
}

ReactDOM.render(<App />, document.getElementById('element'));
```


INDEX.TSX

```
// import the datePickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// creates a datePicker with strictMode property
export default class App extends React.Component<{}>, {}> {
    // sets the value
    private dateValue:Date= new Date('5/28/2017');
    // sets the min
    private minDate:Date= new Date('5/5/2017');
    // sets the max
    private maxDate:Date= new Date('5/25/2017');
    private disable: boolean = true;
    public render() {
        return <DatePickerComponent id="datepicker" strictMode={this.disable}
format="dd/MM/yyyy" value={this.dateValue} min={this.minDate}
max={this.maxDate} placeholder="Enter date" />
    }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]**INDEX.JSX**

```
// import the datePickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// creates a datePicker with strictMode property
function App() {
    // sets the value
    const dateValue = new Date('5/28/2017');
    // sets the min
    const minDate = new Date('5/5/2017');
    // sets the max
    const maxDate = new Date('5/25/2017');
    const disable = true;
    return <DatePickerComponent id="datepicker" strictMode={disable}
format="dd/MM/yyyy" value={dateValue} min={minDate} max={maxDate}
placeholder="Enter date"/>;
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datePickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// creates a datePicker with strictMode property
```

```
function App() {
  // sets the value
  const dateValue:Date= new Date('5/28/2017');
  // sets the min
  const minDate:Date= new Date('5/5/2017');
  // sets the max
  const maxDate:Date= new Date('5/25/2017');
  const disable: boolean = true;
  return <DatePickerComponent id="datepicker" strictMode={disable}
format="dd/MM/yyyy" value={dateValue} min={minDate} max={maxDate}
placeholder="Enter date" />
};
ReactDOM.render(<App />, document.getElementById('element'));
```

By default, the DatePicker act in strictMode **false** state, that allows to enter the invalid or out-of-range date in textbox.

If the date is out-of-range or invalid, then the model value will be set to **out of range** date value or **null** respectively with highlighted **error** class to indicates the date is out of range or invalid.

The following example demonstrates the strictMode as **false**. Here, it allows to enter the valid or invalid value in textbox. If you are entering out-of-range or invalid date value, then the model value will be set to **out of range** date value or **null** respectively with highlighted **error** class to indicates the date is out of range or invalid.

[Class-component]

INDEX.JSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// creates a datepicker with strictMode property
export default class App extends React.Component {
  // sets the value
  dateValue = new Date('5/28/2017');
  // sets the min
  minDate = new Date('5/5/2017');
  // sets the max
  maxDate = new Date('5/25/2017');
  disable = false;
  render() {
    return <DatePickerComponent id="datepicker" strictMode={this.disable}
format="dd/MM/yyyy" value={this.dateValue} min={this.minDate}
max={this.maxDate} placeholder="Enter date"/>;
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
```

```
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// creates a datepicker with strictMode property
export default class App extends React.Component<{}>, {}> {
    // sets the value
    private dateValue:Date= new Date('5/28/2017');
    // sets the min
    private minDate:Date= new Date('5/5/2017');
    // sets the max
    private maxDate:Date= new Date('5/25/2017');
    private disable: boolean = false;
    public render() {
        return <DatePickerComponent id="datepicker" strictMode={this.disable}
format="dd/MM/yyyy" value={this.dateValue} min={this.minDate}
max={this.maxDate} placeholder="Enter date" />
    }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]

INDEX.JSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// creates a datepicker with strictMode property
function App() {
    // sets the value
    const dateValue = new Date('5/28/2017');
    // sets the min
    const minDate = new Date('5/5/2017');
    // sets the max
    const maxDate = new Date('5/25/2017');
    const disable = false;
    return <DatePickerComponent id="datepicker" strictMode={disable}
format="dd/MM/yyyy" value={dateValue} min={minDate} max={maxDate}
placeholder="Enter date"/>;
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// creates a datepicker with strictMode property
function App() {
    // sets the value
    const dateValue:Date= new Date('5/28/2017');
    // sets the min
    const minDate:Date= new Date('5/5/2017');
    // sets the max
```

```
const maxDate:Date= new Date('5/25/2017');
const disable: boolean = false;
return <DatePickerComponent id="datepicker" strictMode={disable}
format="dd/MM/yyyy" value={dateValue} min={minDate} max={maxDate}
placeholder="Enter date" />
};
ReactDOM.render(<App />, document.getElementById('element'));
```

If the value of `min` or `max` properties changed through code behind. Then you have to update the `value` property to set within the range.

Customization in React Datepicker component

You can customize the entire appearance of the input element and Calendar by using custom [cssClass](#) property and also you can use the calendar's [renderDayCell](#) event to customize the appearance of the each day cell.

Below list of available classes are used to customize the entire DatePicker component.

Class Name	Description
---	---
e-date-wrapper	Applied to DatePicker wrapper
e-datepicker	Applied to the DatePicker element.
e-float-text	Applied to the floating label.
e-date-icon	Applied to the DatePicker icon.
e-popup-wrapper	Applied to DatePicker popup wrapper.
e-calendar	Applied to Calendar element.
e-header	Applied to Calendar header.
e-title	Applied to Calendar title.
e-icon-container	Applied to Calendar previous and next icon container.
e-prev	Applied to Calendar previous icon.
e-next	Applied to Calendar next icon.
e-weekend	Applied to Calendar weekend dates.
e-other-month	Applied to Calendar other month dates.
e-day	Applied to each day cell of the Calendar.
e-selected	Applied to Calendar selected dates.
e-disabled	Applied to Calendar disabled dates.

The following example highlights the textbox and calendars's weekend days by using custom CSS. Here we have used the `e-custom-style` class to highlight the textbox and disabled date and `renderDayCell` event to disable the weekends of every month.

[Class-component]

INDEX.JSX

```
// import the datePickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  onRenderDayCell(args) {
    if (args.date.getDay() === 0 || args.date.getDay() === 6) {
      // sets isDisabled to true to disable the date.
      args.isDisabled = true;
      // To know about the disabled date customization, you can refer
      in "styles.css".
    }
  }
  render() {
    return (
      // specifies the tag for render the DatePicker component
      <DatePickerComponent id="datepicker" cssClass="e-custom-style"
      renderDayCell={this.onRenderDayCell} placeholder="Enter date"/>);
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datePickercomponent
import { DatePickerComponent, RenderDayCellEventArgs } from '@syncfusion/ej2-
react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
  public onRenderDayCell(args: RenderDayCellEventArgs): void {
    if ((args.date as Date).getDay() === 0 || (args.date as
    Date).getDay() === 6) {
      // sets isDisabled to true to disable the date.
      args.isDisabled = true;
      // To know about the disabled date customization, you can refer
      in "styles.css".
    }
  }
  public render() {
    return (
      // specifies the tag for render the DatePicker component
      <DatePickerComponent id="datepicker" cssClass="e-custom-style"
      renderDayCell={this.onRenderDayCell} placeholder="Enter date"/>
    );
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]**INDEX.JSX**

```
// import the datePickercomponent
```

```
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    function onRenderDayCell(args) {
        if (args.date.getDay() === 0 || args.date.getDay() === 6) {
            // sets isDisabled to true to disable the date.
            args.isDisabled = true;
            // To know about the disabled date customization, you can refer
            in "styles.css".
        }
    }
    return (
        // specifies the tag for render the DatePicker component
        <DatePickerComponent id="datepicker" cssClass="e-custom-style"
        renderDayCell={onRenderDayCell} placeholder="Enter date"/>);
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DatePickerComponent, RenderDayCellEventArgs } from '@syncfusion/ej2-
react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
    function onRenderDayCell(args: RenderDayCellEventArgs): void {
        if ((args.date as Date).getDay() === 0 || (args.date as
        Date).getDay() === 6) {
            // sets isDisabled to true to disable the date.
            args.isDisabled = true;
            // To know about the disabled date customization, you can refer
            in "styles.css".
        }
    }
    return (
        // specifies the tag for render the DatePicker component
        <DatePickerComponent id="datepicker" cssClass="e-custom-style"
        renderDayCell={onRenderDayCell} placeholder="Enter date"/>
    );
}
ReactDOM.render(<App />, document.getElementById('element'));
```

Adding mandatory asterisk to placeholder and float label

You can add a mandatory asterisk(*) to placeholder and float label using `.e-input-group.e-control-wrapper.e-float-input .e-float-text::after` class.

[Class-component]

INDEX.JSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
```

```
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  render() {
    return <DatePickerComponent id="datepicker" floatLabelType="Auto"
placeholder="Enter date"/>;
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
  public render() {
    return <DatePickerComponent id="datepicker" floatLabelType="Auto"
placeholder="Enter date"/>;
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]

INDEX.JSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  return <DatePickerComponent id="datepicker" floatLabelType="Auto"
placeholder="Enter date"/>;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  return <DatePickerComponent id="datepicker" floatLabelType="Auto"
placeholder="Enter date"/>;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

See Also

- [How to disable the DatePicker component](#)
- [How to set read-only for DatePicker](#)

- [How to customize the DatePicker day header](#)

Date views in React DatePicker component

The DatePicker has the following predefined views that provides a flexible way to navigate back and forth to select the date.

View	Description
--- ---	
month (default)	Displays the days in a month
year	Displays the months in a year
decade	Displays the years in a decade

Start view

You can use the [start](#) property to define the initial rendering view.

The following example demonstrates how to create a DatePicker with [decade](#) as initial rendering view.

[Class-component]

INDEX.JSX

```
// import the datePickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// creates a datePicker with year view.
export default class App extends React.Component {
  render() {
    return <DatePickerComponent id="datepicker" start="Decade"
placeholder="Enter date"/>;
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datePickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// creates a datePicker with year view.
export default class App extends React.Component<{}, {}> {
  public render() {
    return <DatePickerComponent id="datepicker" start="Decade"
placeholder="Enter date"/>;
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]

INDEX.JSX


```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// creates a datepicker with year view.
function App() {
    return <DatePickerComponent id="datepicker" start="Decade"
placeholder="Enter date"/>;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// creates a datepicker with year view.
function App() {
    return <DatePickerComponent id="datepicker" start="Decade"
placeholder="Enter date"/>;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

Depth view

Define the [depth](#) property to control the view navigation.

Always the depth view has to be smaller than the start view, otherwise the view restriction will be not restricted.

The following example demonstrates how to create a DatePicker that allows users to select a month.

[Class-component]**INDEX.JSX**

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// creates a datepicker with decade view and navigate up to year view.
export default class App extends React.Component {
    render() {
        return <DatePickerComponent id="datepicker" start="Decade"
depth="Year" placeholder="Enter date"/>;
    }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// creates a datepicker with decade view and navigate up to year view.
```

```
export default class App extends React.Component<{}, {}> {
  public render() {
    return <DatePickerComponent id="datepicker" start="Decade"
    depth="Year" placeholder="Enter date"/>;
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]

INDEX.JSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// creates a datepicker with decade view and navigate up to year view.
function App() {
  return <DatePickerComponent id="datepicker" start="Decade" depth="Year"
  placeholder="Enter date"/>;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
// creates a datepicker with decade view and navigate up to year view.
function App() {
  return <DatePickerComponent id="datepicker" start="Decade" depth="Year"
  placeholder="Enter date"/>;
}
ReactDOM.render(<App />, document.getElementById('element'));
```

To know more about Calendar views refer the Calendar's [Calendar Views](#) section.

Accessibility in React DatePicker component

The DatePicker component followed the accessibility guidelines and standards, including [ADA](#), [Section 508](#), [WCAG 2.2](#) standards, and [WCAG roles](#) that are commonly used to evaluate accessibility.

The accessibility compliance for the DatePicker component is outlined below.

| Accessibility Criteria | Compatibility |

| -- | -- |

| [WCAG 2.2 Support](#) | |

| [Section 508 Support](#) | |

| [Screen Reader Support](#) | |

```

| Right-To-Left Support |  |
| Color Contrast |  |
| Mobile Device Support |  |
| Keyboard Navigation Support |  |
| Accessibility Checker Validation |  |
| Axe-core Accessibility Validation |  |

<style>
.post .post-content img {
display: inline-block;
margin: 0.5em 0;
}
</style>

<div> - All
features of the component meet the requirement.</div>

<div> - Some features of the component do not meet the requirement.</div>

<div> - The component does not meet the requirement.</div>

```

WAI-ARIA attributes

The Web accessibility defines a way to make web content and web applications more accessible to disabled people. It especially helps the dynamic content change and advanced user interface controls developed with Ajax, HTML, JavaScript, and related technologies.

DatePicker provides built-in compliance with the [WAI-ARIA](#) specifications. WAI-ARIA supports is achieved through the attributes like `aria-expanded`, `aria-disabled`, `aria-activedescendant` applied to the input element.

To know about the accessibility of Calendar refer to the Calendar's [Accessibility](#) section.

It helps to provide information about the widget for assistive technology to the disabled person in screen reader.

- **Aria-expanded:** attributes indicates the state of a collapsible element.
- **Aria-disabled:** attribute indicates the disabled state of this DatePicker component.
- **Aria-activedescendant:** attribute helps in managing the current active child of the DatePicker component.

Keyboard Interaction

You can use the following keys to interact with the DatePicker. The component implements the keyboard navigation support by following the [WAI-ARIA practices](#).

It supports the below list of shortcut keys.

Input Navigation

Before opening the popup, use the below list of keys to control the popup element.

| **Press** | **To do this** |

| --- | --- |

| **Alt + Down Arrow** | Opens the popup. |

| **Alt + Up Arrow** | Closes the popup. |

| **Esc** | closes the popup. |

Calendar Navigation

Use the below list of keys to navigate the Calendar after the popup has opened.

| **Press** | **To do this** |

| --- | --- |

| **Upper Arrow** | Focus the previous week date. |

| **Down Arrow** | Focus the next week date. |

| **Left Arrow** | Focus the previous date. |

| **Right Arrow** | Focus the next date. |

| **Home** | Focus the first date in the month. |

| **End** | Focus the last date in the month. |

| **Page Up** | Focus the same date in the previous month. |

| **Page Down** | Focus the same date in the next month. |

| **Enter** | Select the currently focused date. |

| **Shift + Page Up** | Focus the same date in the previous year. |

| **Shift + Page Down** | Focus the same date in the next year. |

| **Control + Upper Arrow** | Moves into the inner level of view like month-year, year-decade |

| **Control + Down Arrow** | Moves out from the depth level view like decade-year, year-month |

| **Control + Home** | Focus the starting date in the current year. |

| **Control + End** | Focus the ending date in the current year. |

To focus the DatePicker component use the **alt+t** keys.

[Class-component]

INDEX.JSX

```
{% raw %}
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  datepickerInstance;
  componentDidMount() {
    const proxy = this;
    this.datepickerInstance.placeholder = 'Enter Date';
    this.datepickerInstance.dataBind();
    document.onkeyup = (e) => {
      if (e.altKey && e.keyCode === 84 /* t */) {
        // press alt+t to focus the control.
        proxy.datepickerInstance.focusIn();
      }
    };
  }
  render() {
    return (
      // specifies the tag for render the DatePicker component
      <DatePickerComponent id="datepicker" ref={(scope) => {
this.datepickerInstance = scope; }}/>);
    )
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
{% raw %}
import { DatePicker, DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
  private datepickerInstance: DatePicker;
  public componentDidMount() {
    const proxy: any = this;
    this.datepickerInstance.placeholder = 'Enter Date';
    this.datepickerInstance.dataBind();
    document.onkeyup = (e) => {
      if (e.altKey && e.keyCode === 84 /* t */) {
        // press alt+t to focus the control.
        proxy.datepickerInstance.focusIn();
      }
    };
  }
  public render() {
    return (
      // specifies the tag for render the DatePicker component
      <DatePickerComponent id="datepicker" ref={(scope) => {
(this.datepickerInstance as DatePicker | null) = scope; }} />
    );
  }
};
```

```

    }
  };
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}

```

[Functional-component]

INDEX.JSX

```

{% raw %}
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  let datePickerInstance;
  React.useEffect(() => {
    const proxy = this;
    this.datepickerInstance.placeholder = 'Enter Date';
    this.datepickerInstance.dataBind();
    document.onkeyup = (e) => {
      if (e.altKey && e.keyCode === 84 /* t */) {
        // press alt+t to focus the control.
        proxy.datepickerInstance.focusIn();
      }
    };
  }, []);
  return (
    // specifies the tag for render the DatePicker component
    <DatePickerComponent id="datepicker" ref={(scope) => { datePickerInstance
= scope; }}/>);
  )
};
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import { DatePicker, DatePickerComponent } from '@syncfusion/ej2-react-
calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
function App() {
  let datePickerInstance: DatePicker;
  React.useEffect(() => {
    datePickerInstance.placeholder = 'Enter Date';
    datePickerInstance.dataBind();
    document.onkeyup = (e) => {
      if (e.altKey && e.keyCode === 84 /* t */) {
        // press alt+t to focus the control.
        datePickerInstance.focusIn();
      }
    };
  }, []);
  return (
    // specifies the tag for render the DatePicker component

```

```

    <DatePickerComponent id="datepicker" ref={(scope) => {
    (datepickerInstance as DatePicker | null) = scope; }} />
    );
  };
  ReactDOM.render(<App />, document.getElementById('element'));
  {% endraw %}

```

Ensuring accessibility

The DatePicker component's accessibility levels are ensured through an [accessibility-checker](#) and [axe-core](#) software tools during automated testing.

The accessibility compliance of the DatePicker component is shown in the following sample. Open the [sample](#) in a new window to evaluate the accessibility of the DatePicker component with accessibility tools.

See also

- [Accessibility in Syncfusion React components](#)

Style appearance in React Datepicker component

The following content provides the exact CSS structure that can be used to modify the control's appearance based on the user preference.

Customizing the appearance of DatePicker wrapper element

Use the following CSS to customize the appearance of wrapper element.

```
`css
```

/ To specify height and font size /

```
.e-input-group input.e-input, .e-input-group.e-control-wrapper input.e-input {
height: 40px;
font-size: 20px;
}
`
```

Customizing the DatePicker icon element

Use the following CSS to customize the DatePicker icon element

```
`css
```

/ To specify background color and font size /

```
.e-input-group .e-input-group-icon:last-child, .e-input-group.e-control-wrapper .e-input-group-icon:last-child {
font-size: 12px;
background-color: darkgray;
}
`
```

Customizing the Calendar popup of the DatePicker

Please check the below section, to customize the style and appearance of the Calendar component.

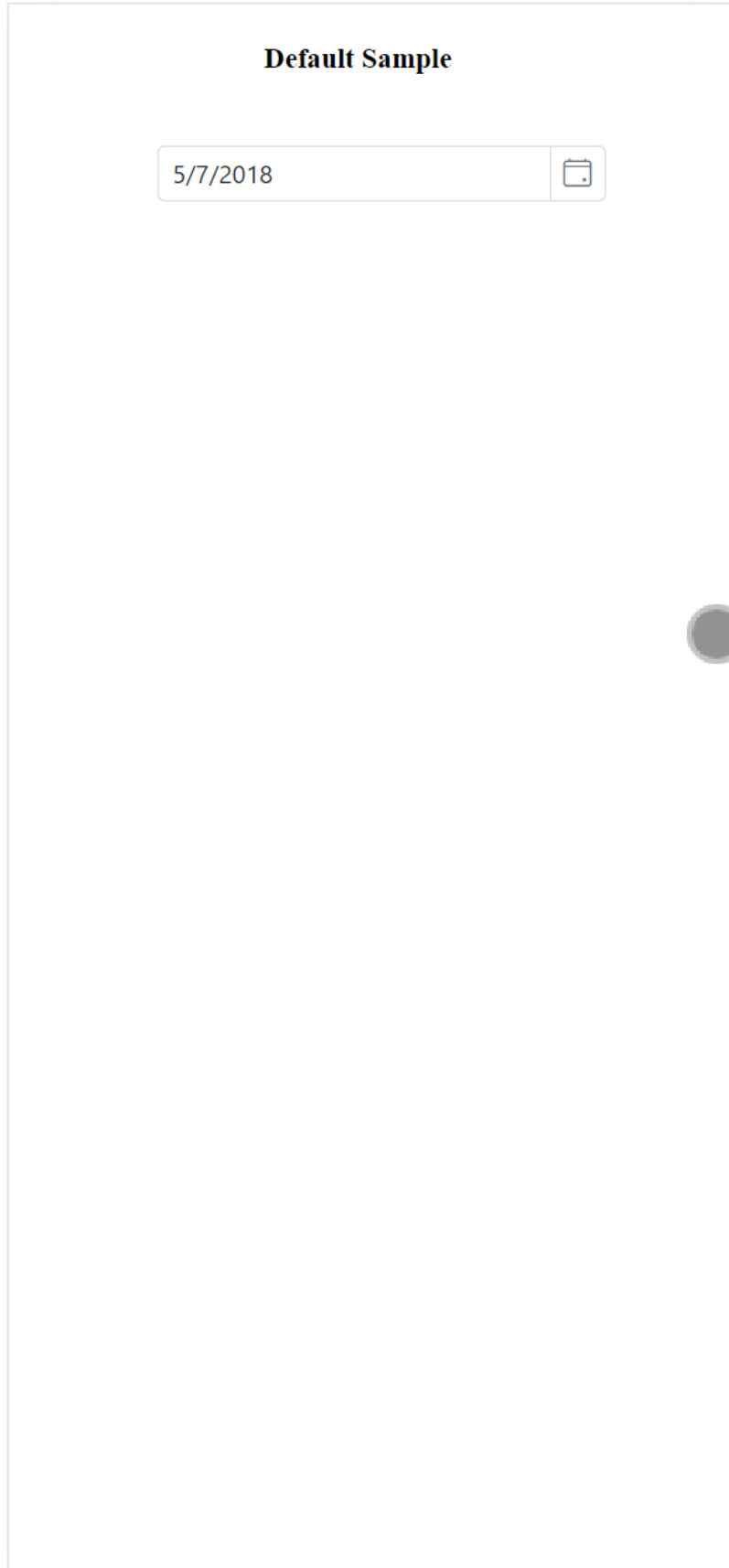
[Customizing Calendar's style and appearance](#)

Full screen mode support in mobiles and tablets

The DatePicker component's full-screen mode feature enables users to view the component popup element in full-screen mode on mobile devices with improved visibility and a better user experience. It is important to mention that this feature is exclusively available for mobile and tablet devices in both landscape and portrait orientations. To activate the full screen mode within the DatePicker component, simply set the [fullScreenMode](#) API value to `true`. This action will extend the calendar element to occupy the entire screen on mobile devices.

`typescript

```
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
  private mobileMode:boolean = true;
  public render() {
    return <DatePickerComponent id="datepicker" fullScreenMode={this.mobileMode} placeholder='Enter date' />
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

How To

Disabled the datepicker component in React Datepicker component

To disable the DatePicker, use its [enable](#) property.

The following example demonstrates the DatePicker in a disabled state.

INDEX.JSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  enable = false;
  render() {
    return (
      // specifies the tag for render the DatePicker component
      <DatePickerComponent enabled={this.enable} placeholder="Enter
date"/>);
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
  private enable:boolean = false;
  public render() {
    return (
      // specifies the tag for render the DatePicker component
      <DatePickerComponent enabled={this.enable} placeholder="Enter
date"/>
    );
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

Customize the datepicker day header in React Datepicker component

You can change the format of the day that to be displayed in header using [dayHeaderFormat](#) property.

By default, the format is **Short**.

You can find the possible formats on below.

Name	Description
----- -----	
Short	Sets the short format of day name (like Su) in day header.
Narrow	Sets the single character of day name (like S) in day header.
Abbreviated	Sets the min format of day name (like Sun) in day header.

| **Wide** | Sets the long format of day name (like Sunday) in day header. |

INDEX.JSX

```
{% raw %}
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  datepickerObj;
  floatLabelObj;
  floatData;
  fields;
  value = 'Short';
  constructor(props) {
    super(props);
    this.floatData = [
      { Id: 'Short', Label: 'Short' },
      { Id: 'Narrow', Label: 'Narrow' },
      { Id: 'Abbreviated', Label: 'Abbreviated' },
      { Id: 'Wide', Label: 'Wide' }
    ];
    this.fields = { text: 'Label', value: 'Id' };
  }
  formatHandler(args) {
    this.datepickerObj.dayHeaderFormat = args.value;
  }
  render() {
    return (
      // specifies the tag for render the DatePicker component
      <div id='container'>
        <div id='datepicker'>
          <DatePickerComponent ref={(datepicker) =>
this.datepickerObj = datepicker} dayHeaderFormat="Short"/>
        </div>
        <div id="format">
          <label className="custom-input-label">Header Format
Types</label>
          <DropDownListComponent id="select" value={this.value}
dataSource={this.floatData} ref={(dropdownlist) => { this.floatLabelObj =
dropdownlist; }} fields={this.fields}
change={this.formatHandler.bind(this)} />
        </div>
      </div>);
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
```

```

import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    public datePickerObj: DatePickerComponent;
    public floatLabelObj: DropDownListComponent;
    private floatData: { [key: string]: Object }[];
    private fields: object;
    private value: string = 'Short';
    constructor(props: {}) {
        super(props);
        this.floatData = [
            { Id: 'Short', Label: 'Short' },
            { Id: 'Narrow', Label: 'Narrow' },
            { Id: 'Abbreviated', Label: 'Abbreviated' },
            { Id: 'Wide', Label: 'Wide' }
        ];
        this.fields = { text: 'Label', value: 'Id' };
    }
    private formatHandler(args: any): void {
        this.datepickerObj.dayHeaderFormat = args.value;
    }
    public render() {
        return (
            // specifies the tag for render the DatePicker component
            <div id='container'>
                <div id='datepicker'>
                    <DatePickerComponent ref={(datepicker) =>
this.datepickerObj = datepicker} dayHeaderFormat="Short"/>
                </div>
                <div id="format">
                    <label className="custom-input-label">Header Format
Types</label>
                    <DropDownListComponent id="select" value = {this.value}
dataSource={this.floatData} ref={(dropdownlist) => { this.floatLabelObj =
dropdownlist }} fields={this.fields} change={this.formatHandler.bind(this)} />
                </div>
            </div>
        );
    }
}
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}

```

Set the placeholder in React Datepicker component

The following example demonstrates how to set `placeholder` in the DatePicker component.

Using `placeholder` you can display a short hint in the input element.

INDEX.JSX

```

// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    render() {

```

```

        return (
            // specifies the tag for render the DatePicker component
            <DatePickerComponent placeholder="Choose a date"/>);
    }
}
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    public render() {
        return (
            // specifies the tag for render the DatePicker component
            <DatePickerComponent placeholder="Choose a date"/>
        );
    }
}
ReactDOM.render(<App />, document.getElementById('element'));

```

Set the readonly in React DatePicker component

The following example demonstrates how to set **readonly** in DatePicker component. You can achieve this by using **readonly** property.

INDEX.JSX

```

// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    readonly = true;
    render() {
        return (
            // specifies the tag for render the DatePicker component
            <DatePickerComponent readonly={this.readonly} placeholder="Enter
date"/>);
    }
}
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    private readonly: boolean = true;
    public render() {
        return (
            // specifies the tag for render the DatePicker component

```

```

        <DatePickerComponent readonly={this.readonly} placeholder="Enter
date"/>
    );
}
}
ReactDOM.render(<App />, document.getElementById('element'));

```

Open datepicker popup on input click in React DatePicker component

You can open the DatePicker popup on input focus by calling the `show` method in the input `focus` event.

The following example demonstrates how to open the DatePicker popup when the input is focused.

INDEX.JSX

```

{% raw %}
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    onFocus(args) {
        this.dateObj.show();
    }
    render() {
        return (
            // Specifies the tag for render the DatePicker component
            <DatePickerComponent focus={this.onFocus.bind(this)}
placeholder="Choose a date" ref={scope => { this.dateObj = scope; }}/>);
        )
    }
}
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
    public onFocus(args: FocusEventArgs) {
        this.dateObj.show();
    }
    public render() {
        return (
            // Specifies the tag for render the DatePicker component
            <DatePickerComponent focus={this.onFocus.bind(this)}
placeholder="Choose a date" ref = {scope => {this.dateObj = scope }}/>
        );
    }
}
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}

```

Prevent the popup close in React DatePicker component

To prevent the DatePicker popup from closing, use the preventDefault method.

The following example demonstrates how to prevent the popup from closing.

INDEX.JSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
  onClose(args) {
    // prevent the popup close
    args.preventDefault();
  }
  render() {
    return (
      // specifies the tag for render the DatePicker component
      <DatePickerComponent close={this.onClose} placeholder="Enter
date"/>);
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}, {}> {
  public onClose(args: PreventableEventArgs) {
    // prevent the popup close
    args.preventDefault();
  }
  public render() {
    return (
      // specifies the tag for render the DatePicker component
      <DatePickerComponent close={this.onClose} placeholder="Enter
date"/>
    );
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

Dynamic form validation in React DatePicker component

Dynamic form can be very useful and more economical to create the forms based on JSON data and without need for adding/changing any code.

- For the dynamic forms, we can create a new React tsx called `dynamic-form.tsx` which is responsible for rendering a dynamic form based on the Json data.

The following example demonstrates how to create the sign up form dynamically.

DATA.JSX

```
export const data = {
  A_FirstName: {
    floatLabelType: 'Auto',
    key: 'FirstName',
    label: 'First Name',
    placeholder: 'First Name',
    type: 'text',
    validation: {
      required: true
    }
  },
  B_LastName: {
    floatLabelType: 'Auto',
    key: 'LastName',
    label: 'Last Name',
    placeholder: 'Last Name',
    type: 'text',
    validation: {
      required: true
    }
  },
  C_Email_ID: {
    floatLabelType: 'Auto',
    key: 'Email_ID',
    label: 'Mail ID',
    placeholder: 'Email',
    type: 'email',
    validation: {
      required: true
    }
  },
  D_Password: {
    floatLabelType: 'Auto',
    key: 'Password',
    label: 'Password',
    placeholder: 'Password',
    type: 'password',
    validation: {
      required: true
    }
  },
  E_DOB: {
    floatLabelType: 'Auto',
    key: 'DOB',
    label: 'DOB',
    placeholder: 'DOB',
    type: 'date',
    validation: {
      required: true
    },
    width: '250px'
  },
  F_Accept: {
    key: 'Accept',
    label: 'Accept terms and conditions',
  }
}
```



```
        type: 'checkbox',
        validation: {
            required: true
        }
    },
    G_Submit: {
        key: 'Button',
        type: 'submit'
    },
};
```

DATA.TSX

```
export const data = {
    A_FirstName: {
        floatLabelType: 'Auto',
        key: 'FirstName',
        label: 'First Name',
        placeholder: 'First Name',
        type: 'text',
        validation: {
            required: true
        }
    },
    B_LastName: {
        floatLabelType: 'Auto',
        key: 'LastName',
        label: 'Last Name',
        placeholder: 'Last Name',
        type: 'text',
        validation: {
            required: true
        }
    },
    C_Email_ID: {
        floatLabelType: 'Auto',
        key: 'Email_ID',
        label: 'Mail ID',
        placeholder: 'Email',
        type: 'email',
        validation: {
            required: true
        }
    },
    D_Password: {
        floatLabelType: 'Auto',
        key: 'Password',
        label: 'Password',
        placeholder: 'Password',
        type: 'password',
        validation: {
            required: true
        }
    },
    E_DOB: {
        floatLabelType: 'Auto',
```

```

        key: 'DOB',
        label: 'DOB',
        placeholder: 'DOB',
        type: 'date',
        validation: {
            required: true
        },
        width: '250px'
    },
    F_Accept: {
        key: 'Accept',
        label: 'Accept terms and conditions',
        type: 'checkbox',
        validation: {
            required: true
        }
    },
    G_Submit: {
        key: 'Button',
        type: 'submit'
    },
}
}

```

DYNAMICFORM.JSX

```

// import the Button component
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
// import the CheckBox component
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
// import the DatePicker component
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
// import the TextBox component
import { TextBoxComponent } from '@syncfusion/ej2-react-inputs';
import * as React from 'react';
export default class DynamicForm extends React.Component {
    objectProps;
    constructor(props) {
        super(props);
    }
    onSubmit() {
        alert("Form submitted!!!");
    }
    renderStatus(element) {
        switch (element.type) {
            case 'date':
                return <div>
                    <DatePickerComponent type={element.type}
floatLabelType={element.floatLabelType} placeholder={element.placeholder}
width={element.width}/>
                </div>;
            case 'checkbox':
                return <div id="checkBox">
                    <CheckBoxComponent type={element.type} label={element.label}/>
                </div>;
            case 'submit':

```

```

        return <ButtonComponent id={element.type}
type={element.type}> Sign up </ButtonComponent>;
        default:
            return <div id="textValue">
                <TextBoxComponent type={element.type}
floatLabelType={element.floatLabelType} placeholder={element.placeholder}/>
            </div>;
        }
    }
    render() {
        this.objectProps = this.props;
        return (Object.keys(this.objectProps.dataObject).map((key, value) =>
{
            return (<div key={key.toString()}>
                <form onSubmit={this.onSubmit}>
                    {this.renderStatus(this.objectProps.dataObject[key])}
                </form>
            </div>);
        }));
    }
}

```

DYNAMICFORM.TSX

```

// import the Button component
import { ButtonComponent } from '@syncfusion/ej2-react-buttons';
// import the CheckBox component
import { CheckBoxComponent } from '@syncfusion/ej2-react-buttons';
// import the DatePicker component
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
// import the TextBox component
import { TextBoxComponent } from '@syncfusion/ej2-react-inputs';
import * as React from 'react';
export default class DynamicForm extends React.Component<any, any> {
    public objectProps: any;
    constructor(props: any) {
        super(props);
    }
    public onSubmit() {
        alert("Form submitted!!!");
    }
    public renderStatus(element: any) {
        switch (element.type) {
            case 'date':
                return <div>
                    <DatePickerComponent
                        type={element.type}
                        floatLabelType={element.floatLabelType}
                        placeholder={element.placeholder}
                        width={element.width} />
                </div>
            case 'checkbox':
                return <div id = "checkBox">
                    <CheckBoxComponent
                        type={element.type}
                        label={element.label} />
                </div>

```

```

        </div>
        case 'submit':
            return <ButtonComponent id={element.type}
                type={element.type}> Sign up </ButtonComponent>
        default:
            return <div id="textValue">
                <TextBoxComponent
                    type={element.type}
                    floatLabelType={element.floatLabelType}
                    placeholder={element.placeholder} />
            </div>
        }
    }
    public render() {
        this.objectProps = this.props;
        return (
            Object.keys(this.objectProps.dataObject).map((key, value) => {
                return (
                    <div key={key.toString()}>
                        <form onSubmit={this.onSubmit} >
                            { this.renderStatus(this.objectProps.dataObject[key]) }
                        </form>
                    </div>
                );
            })
        );
    }
}

```

INDEX.JSX

```

import { data } from './data';
import DynamicForm from './dynamicform';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component {
    data;
    constructor(props) {
        super(props);
        this.data = data;
    }
    render() {
        return (<div className="wrap">
            <div className="outerbox">
                <div className="formValue">
                    <DynamicForm dataObject={data}/>
                </div>
            </div>
        </div>);
    }
}
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import { data } from './data';

```

```
import DynamicForm from './dynamicform';
import * as React from 'react';
import * as ReactDOM from 'react-dom';
export default class App extends React.Component<{}>, {}> {
  public data : any;
  constructor(props: any) {
    super(props);
    this.data = data;
  }
  public render() {
    return (
      <div className = "wrap">
        <div className= "outerbox">
          <div className = "formValue">
            <DynamicForm dataObject = {data} />
          </div>
        </div>
      </div>
    );
  }
}
ReactDOM.render(<App />, document.getElementById('element'));
```

Ej1 api migration in React Datepicker component

This article describes the API migration process of DatePicker component from Essential JS 1 to Essential JS 2.

Date Selection

{% raw %}

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Setting the value	Property: value	Property: value

Date Format

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Display the date format	Property: dateFormat	Property: format
Day header format	Property: dayHeaderFormat	Not Applicable

Calendar Views

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Start view	Property: startLevel	Property: start
Depth view	Property: depthLevel	Property: depth

Date Range

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Minimum date	Property: minDate	Property: min
Maximum date	Property: maxDate	Property: max

Disabled Dates

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Block-out dates	Property: blackoutDates var blackoutDates = [new Date(2016, 4, 10), new Date(2016, 4, 15), new Date(2016, 4, 20), new Date(2016, 4, 22), new Date(2016, 5, 12), new Date(2016, 5, 24)]	Can be achieved by disableDate(args) { if (args.date.getDay() === 0 args.date.getDay() === 6) { args.isDisabled = true; } }

Customization

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
CSS Class	Property: cssClass	Property: cssClass
Event callback for each cell creation	Not Applicable	Event: renderDayCellonRenderCell() {/ code block */}
Show/Hide the today button	Property: showFooter	Property: showTodayButton
Show/Hide the other month dates	Property: showOtherMonths	Can be achieved by.e-datepicker .e-calendar .e-content tr.e-month-hide, .e-datepicker .e-calendar .e-content td.e-other-month > .e-day { visibility: none; }.e-datepicker .e-calendar .e-content td.e-month-hide, .e-datepicker .e-calendar .e-content td.e-other-month { pointer-events: none; touch-action: none;}
Show/Hide the disabled date	Property: showDisabledRangevar blackoutDates = [new Date(2016, 4, 10), new Date(2016, 4, 15), new Date(2016, 4, 20), new Date(2016, 4, 22), new Date(2016, 5, 12), new Date(2016, 5, 24)]	Not Applicable
Show/Hide the popup button	Property: showPopupButton	Event: FocusonFocus(args) { this.show();}.e-control-wrapper .e-input-group-icon.e-date-icon { display: none;}
Enable/Disable the rounded corner	Property: showRoundedCorner	Can be achieved by.e-control-wrapper.e-customStyle.e-date-wrapper.e-input-group { border-radius: 4px;}
Skip a month	Property: stepMonths	Can be achieved byonOpen(args){ datepicker.navigateTo('Year', new Date("03/18/2028"));}
Show/Hide the tooltip	Property: showTooltip	Not Applicable
Today button text	Property: buttonText	Can be achieved byL10n.load({ 'de': { 'datepicker': { placeholder: 'Wählen Sie ein Datum aus', today: 'heute' } } });`
Display Inline	Property: displayInline	Not Applicable

Enable/Disable the animation	Property: enableAnimation	Not Applicable
Highlight dates	Property: highlightSection	Can be achieved by <code>highlightDate(args) { if (args.date.getDate() === 10) { args.element.classList.add('e-highlightweekend'); }}.e-highlightweekend { background-color: #cfe9f3;}</code>
Highlight weekend	Property: highlightWeekend	Can be achieved by <code>highlightDate(args) { if (args.date.getDay() === 0 args.date.getDay() === 6) { args.element.classList.add('e-highlightweekend'); }}.e-highlightweekend { background-color: #cfe9f3;}</code>
Tooltip format	Property: tooltipFormat	Not Applicable
Special dates	Property: specialDates var specialdate = [{ date: new Date(), tooltip: "In Australia" }]	Can be achieved by <code>customDates(args) { if (args.date.getDate() === 10) { var span = document.createElement('span'); span.setAttribute('class', 'e-icons highlight'); args.element.firstChild.setAttribute('title', 'Birthday !'); args.element.setAttribute('title', 'Birthday !'); args.element.setAttribute('data-val', 'Birthday !'); args.element.appendChild(span); }}</code>
FocusIn event	Event: focusIn function onFocus() { / code block */ }	Event: focusOnFocus() { / code block */ }
FocusOut event	Event: focusOut function onFocus() { / code block */ }	Event: blurOnBlur() { / code block */ }
FocusIn method	Not Applicable	Method: focusIn() <code>create(args){ this.focusIn();}</code>
FocusOut method	Not Applicable	Method: focusOut() <code>create(args){ this.focusIn(); this.focusOut();}</code>
Prevent popup close	Not Applicable	Event: Close <code>onClose(args) { /Triggers when the popup gets close/ args.cancel = true;}</code>
Prevent popup open	Not Applicable	Event: Open <code>onOpen(args) { /Triggers when the popup gets close/ args.cancel = true;}</code>

Accessibility

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Enable/Disable the RTL	Property: enableRTL	Property: enableRtl

Persistence

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Enable/Disable the persistence	Property: enablePersistence	Property: enablePersistence

Validation

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Validation rules	Property: validationRules var validationRules = {required: {true}};\$.validator.setDefaults({ errorClass: 'e-validation-error', errorPlacement: function (error, element) { \$(error).insertAfter(element.closest(".e-widget")); } });	Can be achieved by var options = { rules: { 'datepicker': { required: true } } };var formObject = new ej.inputs.FormValidator('#form-element', options);
Validation message	Property: validationMessage var validationRules = {required: {true}};var validationMessage = {required: "Required value"};\$.validator.setDefaults({ errorClass: 'e-validation-error', errorPlacement: function (error, element) { \$(error).insertAfter(element.closest(".e-widget")); } });	Can be achieved by var options = { rules: { 'datepicker': { required: true } },customPlacement: (inputElement, errorElement) => { inputElement.parentElement.parentElement.appendChild(errorElement);}var formObject = new ej.inputs.FormValidator('#form-element', options);

Common

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Width	Property: width	Property: width
Readonly	Property: readonly	Property: readonly
Show/Hide the clear button	Not Applicable	Property: showClearButton
Height	Property: height	Can be achieved by.e-control-wrapper.e-custom-style.e-date-wrapper.e-input-group { height: 35px;}
Html Attributes	Property: HtmlAttributesvar htmlAttributes = {required:"required"}	Not Applicable
Enable/Disable the week number	Property: weekNumber	Property: weekNumber
Watermark text	Property: watermarkText`	

Globalization

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Locale	Property: locale	Property: locale
First day of week	Property: startDay	Property: firstDayOfWeek

Strict Mode

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Strict mode	Property: enableStrictMode	Property: strictMode

Open and Close

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Close	Event: Closefunction onClose() { / code block */ }	Event: CloseonClose() { / code block */ }
Hide	Method: hide()function onCreate(args) { var dateObject = \$("#datepicker").data("ejDatePicker"); dateObject.show(); dateObject.hide();}	Method: hide()onCreate(args){ this.show(); this.hide();}
Open	Event: Openfunction onOpen() {/ code block */ }	Event: OpenonOpen() {/ code block */ }
Show	Method: show()function onCreate(args) { var dateObject = \$("#datepicker").data("ejDatePicker"); dateObject.show();}	Method: show()create(args){ this.show();}

View Navigation

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Navigate to specific month	Not Applicable	Method: navigateTo()onOpen(args){ this.navigateTo('Year', new Date("03/18/2018"));}
Navigation callback	Event: NavigateonNavigate() {/ code block */ }	Event: NavigatedonNavigated() {/ code block */ }
Enable/Disable the drill down	Property: allowDirllDown Link to the Video	Not Applicable

{% endraw %}

DateRangePicker

Getting started

This section explains you the steps required to create a simple DateRangePicker and demonstrate the basic usage of the DateRangePicker component.

To get start quickly with React DateRangePicker, you can check on this video:

Dependencies

The below list of dependencies are required to use the DateRangePicker component in your application.

```
`javascript
```

```
|-- @syncfusion/ej2-react-calendars
|-- @syncfusion/ej2-react-base
|-- @syncfusion/ej2-base
|-- @syncfusion/ej2-data
|-- @syncfusion/ej2-calendars
|-- @syncfusion/ej2-inputs
|-- @syncfusion/ej2-splitbuttons
|-- @syncfusion/ej2-lists
|-- @syncfusion/ej2-popups
|-- @syncfusion/ej2-buttons
,
```

Installation and configuration

You can use [create-react-app](#) to setup the applications.

To install `create-react-app` run the following command.

,

```
npm install -g create-react-app
```

,

- To setup basic `React` sample use following commands.

,

```
create-react-app quickstart --scripts-version=react-scripts-ts
```

```
cd quickstart
```

,

Adding Syncfusion packages

All the available Essential JS 2 packages are published in [npmjs.com](https://www.npmjs.com) public registry. You can choose the component that you want to install. For this application, we are going to use `DateRangePicker` component.

To install `DateRangePicker` component, use the following command

```
`bash
```

```
npm install @syncfusion/ej2-react-calendars --save
```

,

Adding Style sheet to the Application

To render the `DateRangePicker` component, need to import `DateRangePicker` and its dependent component's styles as given below in `App.css`.

```
`css
```

```
@import "../node_modules/@syncfusion/ej2-base/styles/material.css";
@import "../node_modules/@syncfusion/ej2-buttons/styles/material.css";
@import "../node_modules/@syncfusion/ej2-lists/styles/material.css";
@import "../node_modules/@syncfusion/ej2-inputs/styles/material.css";
@import "../node_modules/@syncfusion/ej2-popups/styles/material.css";
@import "../node_modules/@syncfusion/ej2-react-calendars/styles/material.css";
`
```

Note: If you want to refer the combined component styles, please make use of our [CRG](#) (Custom Resource Generator) in your application.

Adding DateRangePicker component to the Application

- To include the DateRangePicker component in application import the `DateRangePickerComponent` from `ej2-react-calendars` package in `App.tsx`.
- Then add the DateRangePicker component as shown in below code example.

[src/App.tsx]

[Class-component]

```
`ts
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import './App.css';
export default class App extends React.Component<{}, {}> {
  public render() {
    return <DateRangePickerComponent id="daterangepicker" />
  }
};
`
```

[Functional-component]

```
`ts
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import './App.css';
function App() {
  return <DateRangePickerComponent id="daterangepicker" />
};
`
```

Run the application

Now run the **npm start** command in the console, it will run your application and open the browser window.

npm start

The below examples shows the basic DateRangePicker component.

[Class-component]

INDEX.JSX

```
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
  render() {
    return <DateRangePickerComponent id="daterangepicker"
placeholder='Select a range' />;
  }
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
  public render() {
    return <DateRangePickerComponent id="daterangepicker"
placeholder='Select a range' />
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]

INDEX.JSX

```
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  return <DateRangePickerComponent id="daterangepicker" placeholder='Select
a range' />;
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    return <DateRangePickerComponent id="daterangepicker" placeholder='Select
a range' />
};
ReactDOM.render(<App />, document.getElementById('element'));
```

Setting the start and end date in a range

The start and end date in a range can be defined with help of [startDate](#) and [endDate](#) property.

The following example demonstrates, how to set the start date, end date on initializing the DateRangePicker. To know more about range restriction in DateRangePicker, please refer this [page](#).

[Class-component]**INDEX.JSX**

```
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
    start = new Date("10/7/2017");
    end = new Date("11/15/2017");
    render() {
        return <DateRangePickerComponent id="daterangepicker"
placeholder='Select a range' startDate={this.start} endDate={this.end}/>;
    }
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
    private start:Date= new Date("10/7/2017");
    private end:Date= new Date("11/15/2017");
    public render() {
        return <DateRangePickerComponent id="daterangepicker"
placeholder='Select a range' startDate={this.start} endDate={this.end} />
    }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]**INDEX.JSX**

```
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
```

```
import * as ReactDOM from "react-dom";
function App() {
    const start = new Date("10/7/2017");
    const end = new Date("11/15/2017");
    return <DateRangePickerComponent id="daterangepicker" placeholder='Select
a range' startDate={start} endDate={end}/>;
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    const start:Date= new Date("10/7/2017");
    const end:Date= new Date("11/15/2017");
    return <DateRangePickerComponent id="daterangepicker" placeholder='Select
a range' startDate={start} endDate={end} />
};
ReactDOM.render(<App />, document.getElementById('element'));
```

See Also

- [Render DateRangePicker with pre-defined ranges](#)
- [Render DateRangePicker with specific culture](#)

Range selection in React Daterangepicker component

Range selection in a DateRangePicker can be made-to-order with desire restrictions based on application needs.

Restrict the range within a range

You can restrict the minimum and maximum date that can be allowed as start date, end date in a range selection with help of [min](#), [max](#) properties.

- **min** – sets the minimum date that can be selected as startDate.
- **max** – sets the maximum date that can be selected as endDate

In the following sample, you can select a date range from 15th date of this month to 15th date of next month.

[Class-component]**INDEX.JSX**

```
// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
    // creates a daterangepicker with min and max property
    minDate = new Date(new Date().getFullYear(), new Date().getMonth(), 15);
```



```

    maxDate = new Date(new Date().getFullYear(), new Date().getMonth() + 1,
15);
    render() {
        return <DateRangePickerComponent id="daterangepicker"
placeholder='Select a range' min={this.minDate} max={this.maxDate}/>;
    }
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}>, {}> {
    // creates a daterangepicker with min and max property
    private minDate:Date= new Date(new Date().getFullYear(), new
Date().getMonth(), 15);
    private maxDate:Date= new Date(new Date().getFullYear(), new
Date().getMonth()+1, 15);
    public render() {
        return <DateRangePickerComponent id="daterangepicker"
placeholder='Select a range' min={this.minDate} max={this.maxDate} />
    }
};
ReactDOM.render(<App />, document.getElementById('element'));

```

[Functional-component]**INDEX.JSX**

```

// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    // creates a daterangepicker with min and max property
    const minDate = new Date(new Date().getFullYear(), new Date().getMonth(),
15);
    const maxDate = new Date(new Date().getFullYear(), new Date().getMonth()
+ 1, 15);
    return <DateRangePickerComponent id="daterangepicker" placeholder='Select
a range' min={minDate} max={maxDate}/>;
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";

```

```
function App() {
  // creates a daterangepicker with min and max property
  const minDate:Date= new Date(new Date().getFullYear(), new
Date().getMonth(), 15);
  const maxDate:Date= new Date(new Date().getFullYear(), new
Date().getMonth()+1, 15);
  return <DateRangePickerComponent id="daterangepicker" placeholder='Select
a range' min={minDate} max={maxDate} />
};
ReactDOM.render(<App />, document.getElementById('element'));
```

If the value of `min` or `max` properties changed through code behind, then you have to update the `start date`, `end date` property to set within the range. Or else, if the `start` and `end date` is out of specified date range, a validation error class will be appended to the input element. If `strictMode` is enabled, and both the start, end date is lesser than the min date then start and end date will be updated with `min` date. If both the start and end date is higher than the max date then start and end date will be updated with `max` date. Or else, if `startDate` is less than `min` date, `startDate` will be updated with `min` date or if `endDate` is greater than `max` date, `endDate` will be updated with the `max` date.

Range span

Days span between ranges can be limited in order to avoid excess or less days selection towards the required days in a range. In this, minimum and maximum span that can be allowed within the date range can be customized by `minDays`, `maxDays` properties.

- `minDays`- Sets the minimum number of days between start date and end date.
- `maxDays`- Sets the maximum number of days between start date and end date.

In the following sample, the range selection should be greater than 3 days and less than 8 days else it won't set.

[Class-component]

INDEX.JSX

```
// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
  render() {
    return (<DateRangePickerComponent id="daterangepicker"
placeholder='Select a range' minDays={3} maxDays={7}/>);
  }
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
```

```
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}>, {}> {
    public render() {
        return (
            <DateRangePickerComponent id="daterangepicker" placeholder='Select a
range' minDays={3} maxDays={7} />
        )
    }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]

INDEX.JSX

```
// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    return (<DateRangePickerComponent id="daterangepicker"
placeholder='Select a range' minDays={3} maxDays={7}/>);
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    return (
        <DateRangePickerComponent id="daterangepicker" placeholder='Select a
range' minDays={3} maxDays={7} />
    )
};
ReactDOM.render(<App />, document.getElementById('element'));
```

Strict mode

DateRangePicker provides the option to limit the user towards entering the valid date only. With strict mode, you can set only valid selection. Also, If any invalid range is specified then the date range value will reset to previous value.

This restriction can be availed by enabling the strict mode by setting true to [strictMode](#) property.

[Class-component]

INDEX.JSX

```
// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
```

```
export default class App extends React.Component {
  render() {
    return <DateRangePickerComponent id="daterangepicker"
placeholder='Select a range' strictMode={true}/>;
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
  public render() {
    return <DateRangePickerComponent id="daterangepicker"
placeholder='Select a range' strictMode={true} />
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]**INDEX.JSX**

```
// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  return <DateRangePickerComponent id="daterangepicker" placeholder='Select
a range' strictMode={true}/>;
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  return <DateRangePickerComponent id="daterangepicker" placeholder='Select
a range' strictMode={true} />
};
ReactDOM.render(<App />, document.getElementById('element'));
```

Globalization in React Daterangepicker component

Globalization is the combination of internalization and localization. You can adapt the component to various languages by parsing and formatting the date or number [Internationalization](#) and also add culture specific customization and translation to the text [localization](#).

By default, DateRangePicker date format and meridian names are specific to the American English culture. It utilizes the [Essential JavaScript 2 Internationalization](#) package to parse and format the date object based on the culture by using the official [UNICODE CLDR](#) JSON data. It provides the `loadCldr` method to load the culture specific CLDR JSON data. To go with the different culture other than English, follow the below steps.

- Install the `CLDR-Data` package by using the below command (it installs all the CLDR JSON data). To know about CLDR-Data refer the [CLDR-Data](#) link.

```
npm install cldr-data --save
```

Once the package installed, you can find the culture specific JSON data under the location `\node_modules\cldr-data`.

- Now, import the installed CLDR JSON data into the `app.ts` file. To import JSON data we need to install the JSON plugin loader. Here we have used the systemJS JSON plugin loader.

```
npm install systemjs-plugin-json --save-dev
```

- Once the package installed, you can find the culture specific JSON data under the location `\node_modules\cldr-data`.
- Now import the installed CLDR JSON data into the `app.ts` file.
- Now use the [loadCldr](#)

method to load the culture specific CLDR JSON data from the installed location to `app.ts` file.

- DateRangePicker displayed `Sunday` as the first day of week based on default culture ("en-US"). If you want to display the DateRangePicker with loaded culture's first day of week, you need to import `weekdata.json` file from the `cldr-data/supplemental` as given in the code example.

```
`ts
import { loadCldr } from "@syncfusion/ej2-base";
import * as gregorian from 'cldr-data/main/de/ca-gregorian.json';
import * as numbers from 'cldr-data/main/de/numbers.json';
import * as timeZoneNames from 'cldr-data/main/de/timeZoneNames.json';
import * as numberingSystems from 'cldr-data/supplemental/numberingSystems.json';
import * as weekData from 'cldr-data/supplemental/weekData.json'; // To load the culture based first
day of week
```

```
loadCldr(numberingSystems, gregorian, numbers, timeZoneNames, weekData);
```

```
,
```

if you are facing the error `/node_modules/cldr-data/main/de/*.json (1,1): unused expression, expected an assignment or function call` when you are adding the json files to render the culture sample, then add the below configuration in your `tslint.json` file

```
`ts
```

```
"linterOptions": {
```

```
"exclude": [
```

```
"*.json",
```

```
"/*.*.json"
```

```
]
```

```
}
```

```
,
```

The `Localization` library allows you to localize default text content of the DateRangePicker. The DateRangePicker component has static text for **today** feature that can be changed to other cultures (Arabic, Deutsch, French, etc.) by defining the [locale](#) value and translation object.

Locale keywords | Text

placeholder | Hint to describe expected value in input element.

startLabel | Label to represent the start date.

endLabel | Label to represent the end date.

applyText | Text present in the apply button.

cancelText | Text present in the cancel button.

selectedDays | Text to represent selected days.

days | Text represents days.

customRange | Text present in the custom range button in presets container.

- Before changing to a culture other than `English`, ensure that locale text for the concerned culture is loaded through `load` method of `L10n` class.

```
`ts
```

```
//Load the L10n from ej2-base
```

```
import { L10n } from "@syncfusion/ej2-base";
```

```
//load the locale object to set the localized placeholder value
```

```
L10n.load({
```

```
'de': {
```

```

'daterangepicker': {
  placeholder: 'Wählen Sie einen Bereich aus',
  startLabel: 'Wählen Sie Startdatum',
  endLabel: 'Wählen Sie Enddatum',
  applyText: 'Sich bewerben',
  cancelText: 'Stornieren',
  selectedDays: 'Ausgewählte Tage',
  days: 'Tage',
  customRange: 'benutzerdefinierten Bereich'
}
}
});
`

```

- Set the culture by using the `locale` property.

The following example demonstrates the DateRangePicker in `German` culture.

[Class-component]

CA-GREGORIAN.JSON

```

{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 12879 $",
          "_cldrVersion": "30.0.3"
        },
        "language": "de"
      },
      "dates": {
        "calendars": {
          "gregorian": {
            "months": {
              "format": {
                "abbreviated": {
                  "1": "Jan.",
                  "2": "Feb.",
                  "3": "März",
                  "4": "Apr.",
                  "5": "Mai",
                  "6": "Juni",
                  "7": "Juli",
                  "8": "Aug.",
                  "9": "Sep.",
                  "10": "Okt.",

```

```
        "11": "Nov.",
        "12": "Dez."
    },
    "narrow": {
        "1": "J",
        "2": "F",
        "3": "M",
        "4": "A",
        "5": "M",
        "6": "J",
        "7": "J",
        "8": "A",
        "9": "S",
        "10": "O",
        "11": "N",
        "12": "D"
    },
    "wide": {
        "1": "Januar",
        "2": "Februar",
        "3": "März",
        "4": "April",
        "5": "Mai",
        "6": "Juni",
        "7": "Juli",
        "8": "August",
        "9": "September",
        "10": "Oktober",
        "11": "November",
        "12": "Dezember"
    }
},
"stand-alone": {
    "abbreviated": {
        "1": "Jan",
        "2": "Feb",
        "3": "Mär",
        "4": "Apr",
        "5": "Mai",
        "6": "Jun",
        "7": "Jul",
        "8": "Aug",
        "9": "Sep",
        "10": "Okt",
        "11": "Nov",
        "12": "Dez"
    },
    "narrow": {
        "1": "J",
        "2": "F",
        "3": "M",
        "4": "A",
        "5": "M",
        "6": "J",
        "7": "J",
        "8": "A",
        "9": "S",
```



```
        "10": "O",
        "11": "N",
        "12": "D"
    },
    "wide": {
        "1": "Januar",
        "2": "Februar",
        "3": "März",
        "4": "April",
        "5": "Mai",
        "6": "Juni",
        "7": "Juli",
        "8": "August",
        "9": "September",
        "10": "Oktober",
        "11": "November",
        "12": "Dezember"
    }
    },
    "days": {
        "format": {
            "abbreviated": {
                "sun": "So.",
                "mon": "Mo.",
                "tue": "Di.",
                "wed": "Mi.",
                "thu": "Do.",
                "fri": "Fr.",
                "sat": "Sa."
            },
            "narrow": {
                "sun": "S",
                "mon": "M",
                "tue": "D",
                "wed": "M",
                "thu": "D",
                "fri": "F",
                "sat": "S"
            },
            "short": {
                "sun": "So.",
                "mon": "Mo.",
                "tue": "Di.",
                "wed": "Mi.",
                "thu": "Do.",
                "fri": "Fr.",
                "sat": "Sa."
            },
            "wide": {
                "sun": "Sonntag",
                "mon": "Montag",
                "tue": "Dienstag",
                "wed": "Mittwoch",
                "thu": "Donnerstag",
                "fri": "Freitag",
                "sat": "Samstag"
            }
        }
    }
}
```

```

    }
  },
  "stand-alone": {
    "abbreviated": {
      "sun": "So",
      "mon": "Mo",
      "tue": "Di",
      "wed": "Mi",
      "thu": "Do",
      "fri": "Fr",
      "sat": "Sa"
    },
    "narrow": {
      "sun": "S",
      "mon": "M",
      "tue": "D",
      "wed": "M",
      "thu": "D",
      "fri": "F",
      "sat": "S"
    },
    "short": {
      "sun": "So.",
      "mon": "Mo.",
      "tue": "Di.",
      "wed": "Mi.",
      "thu": "Do.",
      "fri": "Fr.",
      "sat": "Sa."
    },
    "wide": {
      "sun": "Sonntag",
      "mon": "Montag",
      "tue": "Dienstag",
      "wed": "Mittwoch",
      "thu": "Donnerstag",
      "fri": "Freitag",
      "sat": "Samstag"
    }
  }
},
"quarters": {
  "format": {
    "abbreviated": {
      "1": "Q1",
      "2": "Q2",
      "3": "Q3",
      "4": "Q4"
    },
    "narrow": {
      "1": "1",
      "2": "2",
      "3": "3",
      "4": "4"
    },
    "wide": {
      "1": "1. Quartal",

```

```

        "2": "2. Quartal",
        "3": "3. Quartal",
        "4": "4. Quartal"
    },
    },
    "stand-alone": {
        "abbreviated": {
            "1": "Q1",
            "2": "Q2",
            "3": "Q3",
            "4": "Q4"
        },
        "narrow": {
            "1": "1",
            "2": "2",
            "3": "3",
            "4": "4"
        },
        "wide": {
            "1": "1. Quartal",
            "2": "2. Quartal",
            "3": "3. Quartal",
            "4": "4. Quartal"
        }
    },
    },
    "dayPeriods": {
        "format": {
            "abbreviated": {
                "midnight": "Mitternacht",
                "am": "vorm.",
                "pm": "nachm.",
                "morning1": "morgens",
                "morning2": "vormittags",
                "afternoon1": "mittags",
                "afternoon2": "nachmittags",
                "evening1": "abends",
                "night1": "nachts"
            },
            "narrow": {
                "midnight": "Mitternacht",
                "am": "vm.",
                "pm": "nm.",
                "morning1": "morgens",
                "morning2": "vormittags",
                "afternoon1": "mittags",
                "afternoon2": "nachmittags",
                "evening1": "abends",
                "night1": "nachts"
            },
            "wide": {
                "midnight": "Mitternacht",
                "am": "vorm.",
                "pm": "nachm.",
                "morning1": "morgens",
                "morning2": "vormittags",
                "afternoon1": "mittags",

```

```

        "afternoon2": "nachmittags",
        "evening1": "abends",
        "night1": "nachts"
    }
},
"stand-alone": {
    "abbreviated": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "Morgen",
        "morning2": "Vormittag",
        "afternoon1": "Mittag",
        "afternoon2": "Nachmittag",
        "evening1": "Abend",
        "night1": "Nacht"
    },
    "narrow": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "Morgen",
        "morning2": "Vormittag",
        "afternoon1": "Mittag",
        "afternoon2": "Nachmittag",
        "evening1": "Abend",
        "night1": "Nacht"
    },
    "wide": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "Morgen",
        "morning2": "Vormittag",
        "afternoon1": "Mittag",
        "afternoon2": "Nachmittag",
        "evening1": "Abend",
        "night1": "Nacht"
    }
}
},
"eras": {
    "eraNames": {
        "0": "v. Chr.",
        "0-alt-variant": "vor unserer Zeitrechnung",
        "1": "n. Chr.",
        "1-alt-variant": "unserer Zeitrechnung"
    },
    "eraAbbr": {
        "0": "v. Chr.",
        "0-alt-variant": "v. u. Z.",
        "1": "n. Chr.",
        "1-alt-variant": "u. Z."
    },
    "eraNarrow": {
        "0": "v. Chr.",
        "0-alt-variant": "v. u. Z.",

```

```

        "l": "n. Chr.",
        "l-alt-variant": "u. Z."
    },
    },
    "dateFormats": {
        "full": "EEEE, d. MMMM y",
        "long": "d. MMMM y",
        "medium": "dd.MM.y",
        "short": "dd.MM.yy"
    },
    "timeFormats": {
        "full": "HH:mm:ss zzzz",
        "long": "HH:mm:ss z",
        "medium": "HH:mm:ss",
        "short": "HH:mm"
    },
    "dateTimeFormats": {
        "full": "{1} 'um' {0}",
        "long": "{1} 'um' {0}",
        "medium": "{1}, {0}",
        "short": "{1}, {0}",
        "availableFormats": {
            "d": "d",
            "E": "ccc",
            "Ed": "E, d.",
            "Ehm": "E h:mm a",
            "EHm": "E, HH:mm",
            "Ehms": "E, h:mm:ss a",
            "EHms": "E, HH:mm:ss",
            "Gy": "y G",
            "GyMMM": "MMM y G",
            "GyMMMd": "d. MMM y G",
            "GyMMMED": "E, d. MMM y G",
            "h": "h 'Uhr' a",
            "H": "HH 'Uhr'",
            "hm": "h:mm a",
            "Hm": "HH:mm",
            "hms": "h:mm:ss a",
            "Hms": "HH:mm:ss",
            "hmsv": "h:mm:ss a v",
            "Hmsv": "HH:mm:ss v",
            "hmv": "h:mm a v",
            "Hmv": "HH:mm v",
            "M": "L",
            "Md": "d.M.",
            "MEd": "E, d.M.",
            "MMd": "d.MM.",
            "MMdd": "dd.MM.",
            "MMM": "LLL",
            "MMMd": "d. MMM",
            "MMMED": "E, d. MMM",
            "MMMMd": "d. MMMM",
            "MMMMEd": "E, d. MMMM",
            "MMMMW": "'Woche' W 'im' MMM",
            "MMMMW": "'Woche' W 'im' MMM",
            "ms": "mm:ss",
            "y": "y",

```

```

        "yM": "M.y",
        "yMd": "d.M.y",
        "yMEd": "E, d.M.y",
        "yMM": "MM.y",
        "yMMdd": "dd.MM.y",
        "yMMM": "MMM y",
        "yMMMd": "d. MMM y",
        "yMMMEd": "E, d. MMM y",
        "yMMMM": "MMMM y",
        "yQQQ": "QQQ y",
        "yQQQQ": "QQQQ y",
        "yw": "'Woche' w 'des' 'Jahres' y",
        "yw": "'Woche' w 'des' 'Jahres' y"
    },
    "appendItems": {
        "Day": "{0} ({2}: {1})",
        "Day-Of-Week": "{0} {1}",
        "Era": "{1} {0}",
        "Hour": "{0} ({2}: {1})",
        "Minute": "{0} ({2}: {1})",
        "Month": "{0} ({2}: {1})",
        "Quarter": "{0} ({2}: {1})",
        "Second": "{0} ({2}: {1})",
        "Timezone": "{0} {1}",
        "Week": "{0} ({2}: {1})",
        "Year": "{1} {0}"
    },
    "intervalFormats": {
        "intervalFormatFallback": "{0} - {1}",
        "d": {
            "d": "d.-d."
        },
        "h": {
            "a": "h 'Uhr' a - h 'Uhr' a",
            "h": "h - h 'Uhr' a"
        },
        "H": {
            "H": "HH-HH 'Uhr'"
        },
        "hm": {
            "a": "h:mm a - h:mm a",
            "h": "h:mm-h:mm a",
            "m": "h:mm-h:mm a"
        },
        "Hm": {
            "H": "HH:mm-HH:mm 'Uhr'",
            "m": "HH:mm-HH:mm 'Uhr'"
        },
        "hmv": {
            "a": "h:mm a - h:mm a v",
            "h": "h:mm-h:mm a v",
            "m": "h:mm-h:mm a v"
        },
        "Hmv": {
            "H": "HH:mm-HH:mm 'Uhr' v",
            "m": "HH:mm-HH:mm 'Uhr' v"
        }
    }
},

```

```

    "hv": {
      "a": "h a - h a v",
      "h": "h-h a v"
    },
    "Hv": {
      "H": "HH-HH 'Uhr' v"
    },
    "M": {
      "M": "M.-M."
    },
    "Md": {
      "d": "dd.MM. - dd.MM.",
      "M": "dd.MM. - dd.MM."
    },
    "MEd": {
      "d": "E, dd.MM. - E, dd.MM.",
      "M": "E, dd.MM. - E, dd.MM."
    },
    "MMM": {
      "M": "MMM-MMM"
    },
    "MMMd": {
      "d": "d.-d. MMM",
      "M": "d. MMM - d. MMM"
    },
    "MMMED": {
      "d": "E, d. - E, d. MMM",
      "M": "E, d. MMM - E, d. MMM"
    },
    "MMMM": {
      "M": "LLLL-LLLL"
    },
    "Y": {
      "Y": "Y-Y"
    },
    "yM": {
      "M": "MM.y - MM.y",
      "Y": "MM.y - MM.y"
    },
    "yMd": {
      "d": "dd.MM.y - dd.MM.y",
      "M": "dd.MM.y - dd.MM.y",
      "Y": "dd.MM.y - dd.MM.y"
    },
    "yMEd": {
      "d": "E, dd.MM.y - E, dd.MM.y",
      "M": "E, dd.MM.y - E, dd.MM.y",
      "Y": "E, dd.MM.y - E, dd.MM.y"
    },
    "yMMM": {
      "M": "MMM-MMM y",
      "Y": "MMM y - MMM y"
    },
    "yMMMd": {
      "d": "d.-d. MMM y",
      "M": "d. MMM - d. MMM y",
      "Y": "d. MMM y - d. MMM y"
    }
  }

```

```
    },  
    "yMMMEd": {  
      "d": "E, d. - E, d. MMM y",  
      "M": "E, d. MMM - E, d. MMM y",  
      "Y": "E, d. MMM y - E, d. MMM y"  
    },  
    "yMMMM": {  
      "M": "MMMM-MMMM y",  
      "Y": "MMMM y - MMMM y"  
    }  
  }  
}  
  
}  
  
}  
  
}  
  
}
```

CA-GREGORIAN.JSX

```
{
  "main";
  {
    "de";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 12879 $",
          "_cldrVersion";
          "30.0.3";
        }
        "language";
        "de";
      }
    }
    "dates";
    {
      "calendars";
      {
        "gregorian";
        {
          "months";
          {
            "format";
            {
              "abbreviated";
              {
                "1";
                "Jan.",
                "2";
                "Feb.",
                "3";
                "März",

```



```
        "4";
        "Apr.",
        "5";
        "Mai",
        "6";
        "Juni",
        "7";
        "Juli",
        "8";
        "Aug.",
        "9";
        "Sep.",
        "10";
        "Okt.",
        "11";
        "Nov.",
        "12";
        "Dez.";
    }
    "narrow";
    {
        "1";
        "J",
        "2";
        "F",
        "3";
        "M",
        "4";
        "A",
        "5";
        "M",
        "6";
        "J",
        "7";
        "J",
        "8";
        "A",
        "9";
        "S",
        "10";
        "O",
        "11";
        "N",
        "12";
        "D";
    }
    "wide";
    {
        "1";
        "Januar",
        "2";
        "Februar",
        "3";
        "März",
        "4";
        "April",
        "5";
```

```
        "Mai",
        "6";
        "Juni",
        "7";
        "Juli",
        "8";
        "August",
        "9";
        "September",
        "10";
        "Oktober",
        "11";
        "November",
        "12";
        "Dezember";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
        "Jan",
        "2";
        "Feb",
        "3";
        "Mär",
        "4";
        "Apr",
        "5";
        "Mai",
        "6";
        "Jun",
        "7";
        "Jul",
        "8";
        "Aug",
        "9";
        "Sep",
        "10";
        "Okt",
        "11";
        "Nov",
        "12";
        "Dez";
    }
    "narrow";
    {
        "1";
        "J",
        "2";
        "F",
        "3";
        "M",
        "4";
        "A",
        "5";
    }
}
```

```

        "M",
        "6";
        "J",
        "7";
        "J",
        "8";
        "A",
        "9";
        "S",
        "10";
        "O",
        "11";
        "N",
        "12";
        "D";
    }
    "wide";
    {
        "1";
        "Januar",
        "2";
        "Februar",
        "3";
        "März",
        "4";
        "April",
        "5";
        "Mai",
        "6";
        "Juni",
        "7";
        "Juli",
        "8";
        "August",
        "9";
        "September",
        "10";
        "Oktober",
        "11";
        "November",
        "12";
        "Dezember";
    }
}
"days";
{
    "format";
    {
        "abbreviated";
        {
            "sun";
            "So.",
            "mon";
            "Mo.",
            "tue";
            "Di.",

```

```
        "wed";
        "Mi.",
        "thu";
        "Do.",
        "fri";
        "Fr.",
        "sat";
        "Sa.";
    }
    "narrow";
    {
        "sun";
        "S",
        "mon";
        "M",
        "tue";
        "D",
        "wed";
        "M",
        "thu";
        "D",
        "fri";
        "F",
        "sat";
        "S";
    }
    "short";
    {
        "sun";
        "So.",
        "mon";
        "Mo.",
        "tue";
        "Di.",
        "wed";
        "Mi.",
        "thu";
        "Do.",
        "fri";
        "Fr.",
        "sat";
        "Sa.";
    }
    "wide";
    {
        "sun";
        "Sonntag",
        "mon";
        "Montag",
        "tue";
        "Dienstag",
        "wed";
        "Mittwoch",
        "thu";
        "Donnerstag",
        "fri";
        "Freitag",
```

```
        "sat";
        "Samstag";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "sun";
        "So",
        "mon";
        "Mo",
        "tue";
        "Di",
        "wed";
        "Mi",
        "thu";
        "Do",
        "fri";
        "Fr",
        "sat";
        "Sa";
    }
    "narrow";
    {
        "sun";
        "S",
        "mon";
        "M",
        "tue";
        "D",
        "wed";
        "M",
        "thu";
        "D",
        "fri";
        "F",
        "sat";
        "S";
    }
    "short";
    {
        "sun";
        "So.",
        "mon";
        "Mo.",
        "tue";
        "Di.",
        "wed";
        "Mi.",
        "thu";
        "Do.",
        "fri";
        "Fr.",
        "sat";
        "Sa.";
    }
}
```

```
        "wide";
        {
            "sun";
            "Sonntag",
            "mon";
            "Montag",
            "tue";
            "Dienstag",
            "wed";
            "Mittwoch",
            "thu";
            "Donnerstag",
            "fri";
            "Freitag",
            "sat";
            "Samstag";
        }
    }
}
"quarters";
{
    "format";
    {
        "abbreviated";
        {
            "1";
            "Q1",
            "2";
            "Q2",
            "3";
            "Q3",
            "4";
            "Q4";
        }
        "narrow";
        {
            "1";
            "1",
            "2";
            "2",
            "3";
            "3",
            "4";
            "4";
        }
        "wide";
        {
            "1";
            "1. Quartal",
            "2";
            "2. Quartal",
            "3";
            "3. Quartal",
            "4";
            "4. Quartal";
        }
    }
}
```

```
        "stand-alone";
        {
            "abbreviated";
            {
                "1";
                "Q1",
                "2";
                "Q2",
                "3";
                "Q3",
                "4";
                "Q4";
            }
            "narrow";
            {
                "1";
                "1",
                "2";
                "2",
                "3";
                "3",
                "4";
                "4";
            }
            "wide";
            {
                "1";
                "1. Quartal",
                "2";
                "2. Quartal",
                "3";
                "3. Quartal",
                "4";
                "4. Quartal";
            }
        }
    }
    "dayPeriods";
    {
        "format";
        {
            "abbreviated";
            {
                "midnight";
                "Mitternacht",
                "am";
                "vorm.",
                "pm";
                "nachm.",
                "morning1";
                "morgens",
                "morning2";
                "vormittags",
                "afternoon1";
                "mittags",
                "afternoon2";
                "nachmittags",
```

```
        "evening1";
        "abends",
        "night1";
        "nachts";
    }
    "narrow";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vm.",
        "pm";
        "nm.",
        "morning1";
        "morgens",
        "morning2";
        "vormittags",
        "afternoon1";
        "mittags",
        "afternoon2";
        "nachmittags",
        "evening1";
        "abends",
        "night1";
        "nachts";
    }
    "wide";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vorm.",
        "pm";
        "nachm.",
        "morning1";
        "morgens",
        "morning2";
        "vormittags",
        "afternoon1";
        "mittags",
        "afternoon2";
        "nachmittags",
        "evening1";
        "abends",
        "night1";
        "nachts";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vorm.",
        "pm";
```



```
        "nachm.",
        "morning1";
    "Morgen",
        "morning2";
    "Vormittag",
        "afternoon1";
    "Mittag",
        "afternoon2";
    "Nachmittag",
        "evening1";
    "Abend",
        "night1";
    "Nacht";
}
"narrow";
{
    "midnight";
    "Mitternacht",
        "am";
    "vorm.",
        "pm";
    "nachm.",
        "morning1";
    "Morgen",
        "morning2";
    "Vormittag",
        "afternoon1";
    "Mittag",
        "afternoon2";
    "Nachmittag",
        "evening1";
    "Abend",
        "night1";
    "Nacht";
}
"wide";
{
    "midnight";
    "Mitternacht",
        "am";
    "vorm.",
        "pm";
    "nachm.",
        "morning1";
    "Morgen",
        "morning2";
    "Vormittag",
        "afternoon1";
    "Mittag",
        "afternoon2";
    "Nachmittag",
        "evening1";
    "Abend",
        "night1";
    "Nacht";
}
}
```

```

    }
    "eras";
    {
      "eraNames";
      {
        "0";
        "v. Chr.",
        "0-alt-variant";
        "vor unserer Zeitrechnung",
        "1";
        "n. Chr.",
        "1-alt-variant";
        "unserer Zeitrechnung";
      }
      "eraAbbr";
      {
        "0";
        "v. Chr.",
        "0-alt-variant";
        "v. u. Z.",
        "1";
        "n. Chr.",
        "1-alt-variant";
        "u. Z.";
      }
      "eraNarrow";
      {
        "0";
        "v. Chr.",
        "0-alt-variant";
        "v. u. Z.",
        "1";
        "n. Chr.",
        "1-alt-variant";
        "u. Z.";
      }
    }
    "dateFormats";
    {
      "full";
      "EEEE, d. MMMM y",
      "long";
      "d. MMMM y",
      "medium";
      "dd.MM.y",
      "short";
      "dd.MM.yy";
    }
    "timeFormats";
    {
      "full";
      "HH:mm:ss zzzz",
      "long";
      "HH:mm:ss z",
      "medium";
      "HH:mm:ss",
      "short";
    }
  }

```

```

        "HH:mm";
    }
    "dateTimeFormats":
    {
        "full";
        "{1} 'um' {0}",
        "long";
        "{1} 'um' {0}",
        "medium";
        "{1}, {0}",
        "short";
        "{1}, {0}",
        "availableFormats":
        {
            "d";
            "d",
            "E";
            "ccc",
            "Ed";
            "E, d.",
            "Ehm";
            "E h:mm a",
            "EHm";
            "E, HH:mm",
            "Ehms";
            "E, h:mm:ss a",
            "EHms";
            "E, HH:mm:ss",
            "Gy";
            "y G",
            "GyMMM";
            "MMM y G",
            "GyMMMd";
            "d. MMM y G",
            "GyMMMED";
            "E, d. MMM y G",
            "h";
            "h 'Uhr' a",
            "H";
            "HH 'Uhr'",
            "hm";
            "h:mm a",
            "Hm";
            "HH:mm",
            "hms";
            "h:mm:ss a",
            "Hms";
            "HH:mm:ss",
            "hmsv";
            "h:mm:ss a v",
            "Hmsv";
            "HH:mm:ss v",
            "hmv";
            "h:mm a v",
            "Hmv";
            "HH:mm v",
            "M";
        }
    }
}

```

```

        "L",
        "Md";
        "d.M.",
        "MEd";
        "E, d.M.",
        "MMd";
        "d.MM.",
        "MMdd";
        "dd.MM.",
        "MMM";
        "LLL",
        "MMMd";
        "d. MMM",
        "MMMEd";
        "E, d. MMM",
        "MMMMd";
        "d. MMMM",
        "MMMMEd";
        "E, d. MMMM",
        "MMMMW";
        "'Woche' W 'im' MMM",
        "MMMMW";
        "'Woche' W 'im' MMM",
        "ms";
        "mm:ss",
        "y";
        "Y",
        "yM";
        "M.y",
        "yMd";
        "d.M.y",
        "yMEd";
        "E, d.M.y",
        "yMM";
        "MM.y",
        "yMMdd";
        "dd.MM.y",
        "yMMM";
        "MMM y",
        "yMMMd";
        "d. MMM y",
        "yMMMEd";
        "E, d. MMM y",
        "yMMMM";
        "MMMM y",
        "yQQQ";
        "QQQ y",
        "yQQQQ";
        "QQQQ y",
        "yw";
        "'Woche' w 'des' 'Jahres' y",
        "yw";
        "'Woche' w 'des' 'Jahres' y";
    }
    "appendItems";
    {
        "Day";

```

```

    "{0} ({2}: {1})",
    "Day-Of-Week";
    "{0} {1}",
    "Era";
    "{1} {0}",
    "Hour";
    "{0} ({2}: {1})",
    "Minute";
    "{0} ({2}: {1})",
    "Month";
    "{0} ({2}: {1})",
    "Quarter";
    "{0} ({2}: {1})",
    "Second";
    "{0} ({2}: {1})",
    "Timezone";
    "{0} {1}",
    "Week";
    "{0} ({2}: {1})",
    "Year";
    "{1} {0}";
  }
  "intervalFormats";
  {
    "intervalFormatFallback";
    "{0} - {1}",
    "d";
    {
      "d";
      "d.-d.";
    }
    "h";
    {
      "a";
      "h 'Uhr' a - h 'Uhr' a",
      "h";
      "h - h 'Uhr' a";
    }
    "H";
    {
      "H";
      "HH-HH 'Uhr'";
    }
    "hm";
    {
      "a";
      "h:mm a - h:mm a",
      "h";
      "h:mm-h:mm a",
      "m";
      "h:mm-h:mm a";
    }
    "Hm";
    {
      "H";
      "HH:mm-HH:mm 'Uhr'",
      "m";
    }
  }

```

```

        "HH:mm-HH:mm 'Uhr'";
    }
    "hmv";
    {
        "a";
        "h:mm a - h:mm a v",
        "h";
        "h:mm-h:mm a v",
        "m";
        "h:mm-h:mm a v";
    }
    "Hmv";
    {
        "H";
        "HH:mm-HH:mm 'Uhr' v",
        "m";
        "HH:mm-HH:mm 'Uhr' v";
    }
    "hv";
    {
        "a";
        "h a - h a v",
        "h";
        "h-h a v";
    }
    "Hv";
    {
        "H";
        "HH-HH 'Uhr' v";
    }
    "M";
    {
        "M";
        "M.-M.";
    }
    "Md";
    {
        "d";
        "dd.MM. - dd.MM.",
        "M";
        "dd.MM. - dd.MM.";
    }
    "MEd";
    {
        "d";
        "E, dd.MM. - E, dd.MM.",
        "M";
        "E, dd.MM. - E, dd.MM.";
    }
    "MMM";
    {
        "M";
        "MMM-MMM";
    }
    "MMMd";
    {
        "d";

```

```

        "d.-d. MMM",
        "M";
        "d. MMM - d. MMM";
    }
    "MMMEd";
    {
        "d";
        "E, d. - E, d. MMM",
        "M";
        "E, d. MMM - E, d. MMM";
    }
    "MMMM";
    {
        "M";
        "LLLL-LLLL";
    }
    "Y";
    {
        "Y";
        "Y-Y";
    }
    "yM";
    {
        "M";
        "MM.y - MM.y",
        "Y";
        "MM.y - MM.y";
    }
    "yMd";
    {
        "d";
        "dd.MM.y - dd.MM.y",
        "M";
        "dd.MM.y - dd.MM.y",
        "Y";
        "dd.MM.y - dd.MM.y";
    }
    "yMEd";
    {
        "d";
        "E, dd.MM.y - E, dd.MM.y",
        "M";
        "E, dd.MM.y - E, dd.MM.y",
        "Y";
        "E, dd.MM.y - E, dd.MM.y";
    }
    "yMMM";
    {
        "M";
        "MMM-MMM Y",
        "Y";
        "MMM Y - MMM Y";
    }
    "yMMMd";
    {
        "d";
        "d.-d. MMM Y",

```

```

    "M";
    "d. MMM - d. MMM y",
      "y";
    "d. MMM y - d. MMM y";
  }
  "yMMMEd";
  {
    "d";
    "E, d. - E, d. MMM y",
      "M";
    "E, d. MMM - E, d. MMM y",
      "y";
    "E, d. MMM y - E, d. MMM y";
  }
  "yMMMM";
  {
    "M";
    "MMMM-MMMM y",
      "y";
    "MMMM y - MMMM y";
  }
}
}
}
}
}
}
}
}
}
}
```

CURRENCIES.JSON

```
{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31"
        },
        "language": "de"
      },
      "numbers": {
        "currencies": {
          "ADP": {
            "displayName": "Andorranische Pesete",
            "displayName-count-one": "Andorranische Pesete",
            "displayName-count-other": "Andorranische Peseten",
            "symbol": "ADP"
          },
          "AED": {
            "displayName": "VAE-Dirham",
            "displayName-count-one": "VAE-Dirham",
            "displayName-count-other": "VAE-Dirham",
            "symbol": "AED"
          }
        }
      }
    }
  }
}
```



```

    "AFA": {
      "displayName": "Afghanische Afghani (1927-2002)",
      "displayName-count-one": "Afghanische Afghani (1927-2002)",
      "displayName-count-other": "Afghanische Afghani (1927-2002)",
      "symbol": "AFA"
    },
    "AFN": {
      "displayName": "Afghanischer Afghani",
      "displayName-count-one": "Afghanischer Afghani",
      "displayName-count-other": "Afghanische Afghani",
      "symbol": "AFN"
    },
    "ALK": {
      "displayName": "Albanischer Lek (1946-1965)",
      "displayName-count-one": "Albanischer Lek (1946-1965)",
      "displayName-count-other": "Albanische Lek (1946-1965)"
    },
    "ALL": {
      "displayName": "Albanischer Lek",
      "displayName-count-one": "Albanischer Lek",
      "displayName-count-other": "Albanische Lek",
      "symbol": "ALL"
    },
    "AMD": {
      "displayName": "Armenischer Dram",
      "displayName-count-one": "Armenischer Dram",
      "displayName-count-other": "Armenische Dram",
      "symbol": "AMD"
    },
    "ANG": {
      "displayName": "Niederländische-Antillen-Gulden",
      "displayName-count-one": "Niederländische-Antillen-Gulden",
      "displayName-count-other": "Niederländische-Antillen-Gulden",
      "symbol": "ANG"
    },
    "AOA": {
      "displayName": "Angolanischer Kwanza",
      "displayName-count-one": "Angolanischer Kwanza",
      "displayName-count-other": "Angolanische Kwanza",
      "symbol": "AOA",
      "symbol-alt-narrow": "Kz"
    },
    "AOK": {
      "displayName": "Angolanischer Kwanza (1977-1990)",
      "displayName-count-one": "Angolanischer Kwanza (1977-1990)",
      "displayName-count-other": "Angolanische Kwanza (1977-1990)",
      "symbol": "AOK"
    },
    "AON": {
      "displayName": "Angolanischer Neuer Kwanza (1990-2000)",
      "displayName-count-one": "Angolanischer Neuer Kwanza (1990-2000)",
      "displayName-count-other": "Angolanische Neue Kwanza (1990-2000)",
      "symbol": "AON"
    },
    "AOR": {

```

```

        "displayName": "Angolanischer Kwanza Reajustado (1995-1999)",
        "displayName-count-one": "Angolanischer Kwanza Reajustado (1995-1999)",
        "displayName-count-other": "Angolanische Kwanza Reajustado (1995-1999)",
        "symbol": "AOR"
    },
    "ARA": {
        "displayName": "Argentinischer Austral",
        "displayName-count-one": "Argentinischer Austral",
        "displayName-count-other": "Argentinische Austral",
        "symbol": "ARA"
    },
    "ARL": {
        "displayName": "Argentinischer Peso Ley (1970-1983)",
        "displayName-count-one": "Argentinischer Peso Ley (1970-1983)",
        "displayName-count-other": "Argentinische Pesos Ley (1970-1983)",
        "symbol": "ARL"
    },
    "ARM": {
        "displayName": "Argentinischer Peso (1881-1970)",
        "displayName-count-one": "Argentinischer Peso (1881-1970)",
        "displayName-count-other": "Argentinische Pesos (1881-1970)",
        "symbol": "ARM"
    },
    "ARP": {
        "displayName": "Argentinischer Peso (1983-1985)",
        "displayName-count-one": "Argentinischer Peso (1983-1985)",
        "displayName-count-other": "Argentinische Peso (1983-1985)",
        "symbol": "ARP"
    },
    "ARS": {
        "displayName": "Argentinischer Peso",
        "displayName-count-one": "Argentinischer Peso",
        "displayName-count-other": "Argentinische Pesos",
        "symbol": "ARS",
        "symbol-alt-narrow": "$"
    },
    "ATS": {
        "displayName": "Österreichischer Schilling",
        "displayName-count-one": "Österreichischer Schilling",
        "displayName-count-other": "Österreichische Schilling",
        "symbol": "öS"
    },
    "AUD": {
        "displayName": "Australischer Dollar",
        "displayName-count-one": "Australischer Dollar",
        "displayName-count-other": "Australische Dollar",
        "symbol": "AU$",
        "symbol-alt-narrow": "$"
    },
    "AWG": {
        "displayName": "Aruba-Florin",
        "displayName-count-one": "Aruba-Florin",
        "displayName-count-other": "Aruba-Florin",
        "symbol": "AWG"
    },
    },

```

```

    "AZM": {
      "displayName": "Aserbaidtschan-Manat (1993-2006)",
      "displayName-count-one": "Aserbaidtschan-Manat (1993-2006)",
      "displayName-count-other": "Aserbaidtschan-Manat (1993-2006)",
      "symbol": "AZM"
    },
    "AZN": {
      "displayName": "Aserbaidtschan-Manat",
      "displayName-count-one": "Aserbaidtschan-Manat",
      "displayName-count-other": "Aserbaidtschan-Manat",
      "symbol": "AZN"
    },
    "BAD": {
      "displayName": "Bosnien und Herzegowina Dinar (1992-1994)",
      "displayName-count-one": "Bosnien und Herzegowina Dinar (1992-
1994)",
      "displayName-count-other": "Bosnien und Herzegowina Dinar (1992-
1994)",
      "symbol": "BAD"
    },
    "BAM": {
      "displayName": "Bosnien und Herzegowina Konvertierbare Mark",
      "displayName-count-one": "Bosnien und Herzegowina Konvertierbare
Mark",
      "displayName-count-other": "Bosnien und Herzegowina
Konvertierbare Mark",
      "symbol": "BAM",
      "symbol-alt-narrow": "KM"
    },
    "BAN": {
      "displayName": "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
      "displayName-count-one": "Bosnien und Herzegowina Neuer Dinar
(1994-1997)",
      "displayName-count-other": "Bosnien und Herzegowina Neue Dinar
(1994-1997)",
      "symbol": "BAN"
    },
    "BBD": {
      "displayName": "Barbados-Dollar",
      "displayName-count-one": "Barbados-Dollar",
      "displayName-count-other": "Barbados-Dollar",
      "symbol": "BBD",
      "symbol-alt-narrow": "$"
    },
    "BDT": {
      "displayName": "Bangladesch-Taka",
      "displayName-count-one": "Bangladesch-Taka",
      "displayName-count-other": "Bangladesch-Taka",
      "symbol": "BDT",
      "symbol-alt-narrow": "ট"
    },
    "BEC": {
      "displayName": "Belgischer Franc (konvertibel)",
      "displayName-count-one": "Belgischer Franc (konvertibel)",
      "displayName-count-other": "Belgische Franc (konvertibel)",
      "symbol": "BEC"
    }

```

```

    },
    "BEF": {
      "displayName": "Belgischer Franc",
      "displayName-count-one": "Belgischer Franc",
      "displayName-count-other": "Belgische Franc",
      "symbol": "BEF"
    },
    "BEL": {
      "displayName": "Belgischer Finanz-Franc",
      "displayName-count-one": "Belgischer Finanz-Franc",
      "displayName-count-other": "Belgische Finanz-Franc",
      "symbol": "BEL"
    },
    "BGL": {
      "displayName": "Bulgarische Lew (1962-1999)",
      "displayName-count-one": "Bulgarische Lew (1962-1999)",
      "displayName-count-other": "Bulgarische Lew (1962-1999)",
      "symbol": "BGL"
    },
    "BGM": {
      "displayName": "Bulgarischer Lew (1952-1962)",
      "displayName-count-one": "Bulgarischer Lew (1952-1962)",
      "displayName-count-other": "Bulgarische Lew (1952-1962)",
      "symbol": "BGK"
    },
    "BGN": {
      "displayName": "Bulgarischer Lew",
      "displayName-count-one": "Bulgarischer Lew",
      "displayName-count-other": "Bulgarische Lew",
      "symbol": "BGN"
    },
    "BGO": {
      "displayName": "Bulgarischer Lew (1879-1952)",
      "displayName-count-one": "Bulgarischer Lew (1879-1952)",
      "displayName-count-other": "Bulgarische Lew (1879-1952)",
      "symbol": "BGJ"
    },
    "BHD": {
      "displayName": "Bahrain-Dinar",
      "displayName-count-one": "Bahrain-Dinar",
      "displayName-count-other": "Bahrain-Dinar",
      "symbol": "BHD"
    },
    "BIF": {
      "displayName": "Burundi-Franc",
      "displayName-count-one": "Burundi-Franc",
      "displayName-count-other": "Burundi-Francs",
      "symbol": "BIF"
    },
    "BMD": {
      "displayName": "Bermuda-Dollar",
      "displayName-count-one": "Bermuda-Dollar",
      "displayName-count-other": "Bermuda-Dollar",
      "symbol": "BMD",
      "symbol-alt-narrow": "$"
    },
    "BND": {

```

```

        "displayName": "Brunei-Dollar",
        "displayName-count-one": "Brunei-Dollar",
        "displayName-count-other": "Brunei-Dollar",
        "symbol": "BND",
        "symbol-alt-narrow": "฿"
    },
    "BOB": {
        "displayName": "Bolivanischer Boliviano",
        "displayName-count-one": "Bolivanischer Boliviano",
        "displayName-count-other": "Bolivianische Bolivianos",
        "symbol": "BOB",
        "symbol-alt-narrow": "Bs"
    },
    "BOL": {
        "displayName": "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-one": "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-other": "Bolivianische Bolivianos (1863-
1963)",
        "symbol": "BOL"
    },
    "BOP": {
        "displayName": "Bolivianischer Peso",
        "displayName-count-one": "Bolivianischer Peso",
        "displayName-count-other": "Bolivianische Peso",
        "symbol": "BOP"
    },
    "BOV": {
        "displayName": "Boliviansiche Mvdol",
        "displayName-count-one": "Boliviansiche Mvdol",
        "displayName-count-other": "Bolivianische Mvdol",
        "symbol": "BOV"
    },
    "BRB": {
        "displayName": "Brasilianischer Cruzeiro Novo (1967-1986)",
        "displayName-count-one": "Brasilianischer Cruzeiro Novo (1967-
1986)",
        "displayName-count-other": "Brasilianische Cruzeiro Novo (1967-
1986)",
        "symbol": "BRB"
    },
    "BRC": {
        "displayName": "Brasilianischer Cruzado (1986-1989)",
        "displayName-count-one": "Brasilianischer Cruzado (1986-1989)",
        "displayName-count-other": "Brasilianische Cruzado (1986-1989)",
        "symbol": "BRC"
    },
    "BRE": {
        "displayName": "Brasilianischer Cruzeiro (1990-1993)",
        "displayName-count-one": "Brasilianischer Cruzeiro (1990-1993)",
        "displayName-count-other": "Brasilianische Cruzeiro (1990-1993)",
        "symbol": "BRE"
    },
    "BRL": {
        "displayName": "Brasilianischer Real",
        "displayName-count-one": "Brasilianischer Real",
        "displayName-count-other": "Brasilianische Real",
        "symbol": "R$",

```

```

        "symbol-alt-narrow": "R$"
    },
    "BRN": {
        "displayName": "Brasilianischer Cruzado Novo (1989-1990)",
        "displayName-count-one": "Brasilianischer Cruzado Novo (1989-
1990) ",
        "displayName-count-other": "Brasilianische Cruzado Novo (1989-
1990) ",
        "symbol": "BRN"
    },
    "BRR": {
        "displayName": "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-one": "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-other": "Brasilianische Cruzeiro (1993-1994)",
        "symbol": "BRR"
    },
    "BRZ": {
        "displayName": "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-one": "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-other": "Brasilianischer Cruzeiro (1942-
1967) ",
        "symbol": "BRZ"
    },
    "BSD": {
        "displayName": "Bahamas-Dollar",
        "displayName-count-one": "Bahamas-Dollar",
        "displayName-count-other": "Bahamas-Dollar",
        "symbol": "BSD",
        "symbol-alt-narrow": "$"
    },
    "BTN": {
        "displayName": "Bhutan-Ngultrum",
        "displayName-count-one": "Bhutan-Ngultrum",
        "displayName-count-other": "Bhutan-Ngultrum",
        "symbol": "BTN"
    },
    "BUK": {
        "displayName": "Birmanischer Kyat",
        "displayName-count-one": "Birmanischer Kyat",
        "displayName-count-other": "Birmanische Kyat",
        "symbol": "BUK"
    },
    "BWP": {
        "displayName": "Botswanischer Pula",
        "displayName-count-one": "Botswanischer Pula",
        "displayName-count-other": "Botswanische Pula",
        "symbol": "BWP",
        "symbol-alt-narrow": "P"
    },
    "BYB": {
        "displayName": "Belarus-Rubel (1994-1999)",
        "displayName-count-one": "Belarus-Rubel (1994-1999)",
        "displayName-count-other": "Belarus-Rubel (1994-1999)",
        "symbol": "BYB"
    },
    "BYN": {
        "displayName": "Weißrussischer Rubel",

```

```

        "displayName-count-one": "Weißrussischer Rubel",
        "displayName-count-other": "Weißrussische Rubel",
        "symbol": "BYN",
        "symbol-alt-narrow": "p."
    },
    "BYR": {
        "displayName": "Weißrussischer Rubel (2000-2016)",
        "displayName-count-one": "Weißrussischer Rubel (2000-2016)",
        "displayName-count-other": "Weißrussische Rubel (2000-2016)",
        "symbol": "BYR"
    },
    "BZD": {
        "displayName": "Belize-Dollar",
        "displayName-count-one": "Belize-Dollar",
        "displayName-count-other": "Belize-Dollar",
        "symbol": "BZD",
        "symbol-alt-narrow": "$"
    },
    "CAD": {
        "displayName": "Kanadischer Dollar",
        "displayName-count-one": "Kanadischer Dollar",
        "displayName-count-other": "Kanadische Dollar",
        "symbol": "CA$",
        "symbol-alt-narrow": "$"
    },
    "CDF": {
        "displayName": "Kongo-Franc",
        "displayName-count-one": "Kongo-Franc",
        "displayName-count-other": "Kongo-Francs",
        "symbol": "CDF"
    },
    "CHE": {
        "displayName": "WIR-Euro",
        "displayName-count-one": "WIR-Euro",
        "displayName-count-other": "WIR-Euro",
        "symbol": "CHE"
    },
    "CHF": {
        "displayName": "Schweizer Franken",
        "displayName-count-one": "Schweizer Franken",
        "displayName-count-other": "Schweizer Franken",
        "symbol": "CHF"
    },
    "CHW": {
        "displayName": "WIR Franken",
        "displayName-count-one": "WIR Franken",
        "displayName-count-other": "WIR Franken",
        "symbol": "CHW"
    },
    "CLE": {
        "displayName": "Chilenischer Escudo",
        "displayName-count-one": "Chilenischer Escudo",
        "displayName-count-other": "Chilenische Escudo",
        "symbol": "CLE"
    },
    "CLF": {
        "displayName": "Chilenische Unidades de Fomento",

```

```

        "displayName-count-one": "Chilenische Unidades de Fomento",
        "displayName-count-other": "Chilenische Unidades de Fomento",
        "symbol": "CLF"
    },
    "CLP": {
        "displayName": "Chilenischer Peso",
        "displayName-count-one": "Chilenischer Peso",
        "displayName-count-other": "Chilenische Pesos",
        "symbol": "CLP",
        "symbol-alt-narrow": "$"
    },
    "CNX": {
        "displayName": "Dollar der Chinesischen Volksbank",
        "displayName-count-one": "Dollar der Chinesischen Volksbank",
        "displayName-count-other": "Dollar der Chinesischen Volksbank",
        "symbol": "CNX"
    },
    "CNY": {
        "displayName": "Renminbi Yuan",
        "displayName-count-one": "Chinesischer Yuan",
        "displayName-count-other": "Renminbi Yuan",
        "symbol": "CN¥",
        "symbol-alt-narrow": "¥"
    },
    "COP": {
        "displayName": "Kolumbianischer Peso",
        "displayName-count-one": "Kolumbianischer Peso",
        "displayName-count-other": "Kolumbianische Pesos",
        "symbol": "COP",
        "symbol-alt-narrow": "$"
    },
    "COU": {
        "displayName": "Kolumbianische Unidades de valor real",
        "displayName-count-one": "Kolumbianische Unidad de valor real",
        "displayName-count-other": "Kolumbianische Unidades de valor
real",
        "symbol": "COU"
    },
    "CRC": {
        "displayName": "Costa-Rica-Colón",
        "displayName-count-one": "Costa-Rica-Colón",
        "displayName-count-other": "Costa-Rica-Colón",
        "symbol": "CRC",
        "symbol-alt-narrow": "₡"
    },
    "CSD": {
        "displayName": "Serbischer Dinar (2002-2006)",
        "displayName-count-one": "Serbischer Dinar (2002-2006)",
        "displayName-count-other": "Serbische Dinar (2002-2006)",
        "symbol": "CSD"
    },
    "CSK": {
        "displayName": "Tschechoslowakische Krone",
        "displayName-count-one": "Tschechoslowakische Kronen",
        "displayName-count-other": "Tschechoslowakische Kronen",
        "symbol": "CSK"
    },
    },

```



```

"CUC": {
  "displayName": "Kubanischer Peso (konvertibel)",
  "displayName-count-one": "Kubanischer Peso (konvertibel)",
  "displayName-count-other": "Kubanische Pesos (konvertibel)",
  "symbol": "CUC",
  "symbol-alt-narrow": "Cub$"
},
"CUP": {
  "displayName": "Kubanischer Peso",
  "displayName-count-one": "Kubanischer Peso",
  "displayName-count-other": "Kubanische Pesos",
  "symbol": "CUP",
  "symbol-alt-narrow": "$"
},
"CVE": {
  "displayName": "Cabo-Verde-Escudo",
  "displayName-count-one": "Cabo-Verde-Escudo",
  "displayName-count-other": "Cabo-Verde-Escudos",
  "symbol": "CVE"
},
"CYP": {
  "displayName": "Zypern-Pfund",
  "displayName-count-one": "Zypern Pfund",
  "displayName-count-other": "Zypern Pfund",
  "symbol": "CYP"
},
"CZK": {
  "displayName": "Tschechische Krone",
  "displayName-count-one": "Tschechische Krone",
  "displayName-count-other": "Tschechische Kronen",
  "symbol": "CZK",
  "symbol-alt-narrow": "Kč"
},
"DDM": {
  "displayName": "Mark der DDR",
  "displayName-count-one": "Mark der DDR",
  "displayName-count-other": "Mark der DDR",
  "symbol": "DDM"
},
"DEM": {
  "displayName": "Deutsche Mark",
  "displayName-count-one": "Deutsche Mark",
  "displayName-count-other": "Deutsche Mark",
  "symbol": "DM"
},
"DJF": {
  "displayName": "Dschibuti-Franc",
  "displayName-count-one": "Dschibuti-Franc",
  "displayName-count-other": "Dschibuti-Franc",
  "symbol": "DJF"
},
"DKK": {
  "displayName": "Dänische Krone",
  "displayName-count-one": "Dänische Krone",
  "displayName-count-other": "Dänische Kronen",
  "symbol": "DKK",
  "symbol-alt-narrow": "kr"
}

```

```

    },
    "DOP": {
      "displayName": "Dominikanischer Peso",
      "displayName-count-one": "Dominikanischer Peso",
      "displayName-count-other": "Dominikanische Pesos",
      "symbol": "DOP",
      "symbol-alt-narrow": "$"
    },
    "DZD": {
      "displayName": "Algerischer Dinar",
      "displayName-count-one": "Algerischer Dinar",
      "displayName-count-other": "Algerische Dinar",
      "symbol": "DZD"
    },
    "ECS": {
      "displayName": "Ecuadorianischer Sucre",
      "displayName-count-one": "Ecuadorianischer Sucre",
      "displayName-count-other": "Ecuadorianische Sucre",
      "symbol": "ECS"
    },
    "ECV": {
      "displayName": "Verrechnungseinheit für Ecuador",
      "displayName-count-one": "Verrechnungseinheiten für Ecuador",
      "displayName-count-other": "Verrechnungseinheiten für Ecuador",
      "symbol": "ECV"
    },
    "EEK": {
      "displayName": "Estnische Krone",
      "displayName-count-one": "Estnische Krone",
      "displayName-count-other": "Estnische Kronen",
      "symbol": "EEK"
    },
    "EGP": {
      "displayName": "Ägyptisches Pfund",
      "displayName-count-one": "Ägyptisches Pfund",
      "displayName-count-other": "Ägyptische Pfund",
      "symbol": "EGP",
      "symbol-alt-narrow": "E£"
    },
    "ERN": {
      "displayName": "Eritreischer Nakfa",
      "displayName-count-one": "Eritreischer Nakfa",
      "displayName-count-other": "Eritreische Nakfa",
      "symbol": "ERN"
    },
    "ESA": {
      "displayName": "Spanische Peseta (A-Konten)",
      "displayName-count-one": "Spanische Peseta (A-Konten)",
      "displayName-count-other": "Spanische Peseten (A-Konten)",
      "symbol": "ESA"
    },
    "ESB": {
      "displayName": "Spanische Peseta (konvertibel)",
      "displayName-count-one": "Spanische Peseta (konvertibel)",
      "displayName-count-other": "Spanische Peseten (konvertibel)",
      "symbol": "ESB"
    },
  },

```

```

"ESP": {
  "displayName": "Spanische Peseta",
  "displayName-count-one": "Spanische Peseta",
  "displayName-count-other": "Spanische Peseten",
  "symbol": "ESP",
  "symbol-alt-narrow": "₧"
},
"ETB": {
  "displayName": "Äthiopischer Birr",
  "displayName-count-one": "Äthiopischer Birr",
  "displayName-count-other": "Äthiopische Birr",
  "symbol": "ETB"
},
"EUR": {
  "displayName": "Euro",
  "displayName-count-one": "Euro",
  "displayName-count-other": "Euro",
  "symbol": "€",
  "symbol-alt-narrow": "€"
},
"FIM": {
  "displayName": "Finnische Mark",
  "displayName-count-one": "Finnische Mark",
  "displayName-count-other": "Finnische Mark",
  "symbol": "FIM"
},
"FJD": {
  "displayName": "Fidschi-Dollar",
  "displayName-count-one": "Fidschi-Dollar",
  "displayName-count-other": "Fidschi-Dollar",
  "symbol": "FJD",
  "symbol-alt-narrow": "$"
},
"FKP": {
  "displayName": "Falkland-Pfund",
  "displayName-count-one": "Falkland-Pfund",
  "displayName-count-other": "Falkland-Pfund",
  "symbol": "FKP",
  "symbol-alt-narrow": "Fl£"
},
"FRF": {
  "displayName": "Französischer Franc",
  "displayName-count-one": "Französischer Franc",
  "displayName-count-other": "Französische Franc",
  "symbol": "FRF"
},
"GBP": {
  "displayName": "Britisches Pfund",
  "displayName-count-one": "Britisches Pfund",
  "displayName-count-other": "Britische Pfund",
  "symbol": "£",
  "symbol-alt-narrow": "£"
},
"GEK": {
  "displayName": "Georgischer Kupon Larit",
  "displayName-count-one": "Georgischer Kupon Larit",
  "displayName-count-other": "Georgische Kupon Larit",

```

```

    "symbol": "GEK"
  },
  "GEL": {
    "displayName": "Georgischer Lari",
    "displayName-count-one": "Georgischer Lari",
    "displayName-count-other": "Georgische Lari",
    "symbol": "GEL",
    "symbol-alt-narrow": "₾",
    "symbol-alt-variant": "₾"
  },
  "GHC": {
    "displayName": "Ghanaischer Cedi (1979–2007)",
    "displayName-count-one": "Ghanaischer Cedi (1979–2007)",
    "displayName-count-other": "Ghanaische Cedi (1979–2007)",
    "symbol": "GHC"
  },
  "GHS": {
    "displayName": "Ghanaischer Cedi",
    "displayName-count-one": "Ghanaischer Cedi",
    "displayName-count-other": "Ghanaische Cedi",
    "symbol": "GHS"
  },
  "GIP": {
    "displayName": "Gibraltar-Pfund",
    "displayName-count-one": "Gibraltar-Pfund",
    "displayName-count-other": "Gibraltar Pfund",
    "symbol": "GIP",
    "symbol-alt-narrow": "£"
  },
  "GMD": {
    "displayName": "Gambia-Dalasi",
    "displayName-count-one": "Gambia-Dalasi",
    "displayName-count-other": "Gambia-Dalasi",
    "symbol": "GMD"
  },
  "GNF": {
    "displayName": "Guinea-Franc",
    "displayName-count-one": "Guinea-Franc",
    "displayName-count-other": "Guinea-Franc",
    "symbol": "GNF",
    "symbol-alt-narrow": "F.G."
  },
  "GNS": {
    "displayName": "Guineischer Syli",
    "displayName-count-one": "Guineischer Syli",
    "displayName-count-other": "Guineische Syli",
    "symbol": "GNS"
  },
  "GQE": {
    "displayName": "Äquatorialguinea-Ekwele",
    "displayName-count-one": "Äquatorialguinea-Ekwele",
    "displayName-count-other": "Äquatorialguinea-Ekwele",
    "symbol": "GQE"
  },
  "GRD": {
    "displayName": "Griechische Drachme",
    "displayName-count-one": "Griechische Drachme",

```

```

        "displayName-count-other": "Griechische Drachmen",
        "symbol": "GRD"
    },
    "GTQ": {
        "displayName": "Guatemaltekischer Quetzal",
        "displayName-count-one": "Guatemaltekischer Quetzal",
        "displayName-count-other": "Guatemaltekische Quetzales",
        "symbol": "GTQ",
        "symbol-alt-narrow": "Q"
    },
    "GWE": {
        "displayName": "Portugiesisch Guinea Escudo",
        "displayName-count-one": "Portugiesisch Guinea Escudo",
        "displayName-count-other": "Portugiesisch Guinea Escudo",
        "symbol": "GWE"
    },
    "GWP": {
        "displayName": "Guinea-Bissau Peso",
        "displayName-count-one": "Guinea-Bissau Peso",
        "displayName-count-other": "Guinea-Bissau Pesos",
        "symbol": "GWP"
    },
    "GYD": {
        "displayName": "Guyana-Dollar",
        "displayName-count-one": "Guyana-Dollar",
        "displayName-count-other": "Guyana-Dollar",
        "symbol": "GYD",
        "symbol-alt-narrow": "$"
    },
    "HKD": {
        "displayName": "Hongkong-Dollar",
        "displayName-count-one": "Hongkong-Dollar",
        "displayName-count-other": "Hongkong-Dollar",
        "symbol": "HK$",
        "symbol-alt-narrow": "$"
    },
    "HNL": {
        "displayName": "Honduras-Lempira",
        "displayName-count-one": "Honduras-Lempira",
        "displayName-count-other": "Honduras-Lempira",
        "symbol": "HNL",
        "symbol-alt-narrow": "L"
    },
    "HRD": {
        "displayName": "Kroatischer Dinar",
        "displayName-count-one": "Kroatischer Dinar",
        "displayName-count-other": "Kroatische Dinar",
        "symbol": "HRD"
    },
    "HRK": {
        "displayName": "Kroatischer Kuna",
        "displayName-count-one": "Kroatischer Kuna",
        "displayName-count-other": "Kroatische Kuna",
        "symbol": "HRK",
        "symbol-alt-narrow": "kn"
    },
    "HTG": {

```

```

        "displayName": "Haitianische Gourde",
        "displayName-count-one": "Haitianische Gourde",
        "displayName-count-other": "Haitianische Gourdes",
        "symbol": "HTG"
    },
    "HUF": {
        "displayName": "Ungarischer Forint",
        "displayName-count-one": "Ungarischer Forint",
        "displayName-count-other": "Ungarische Forint",
        "symbol": "HUF",
        "symbol-alt-narrow": "Ft"
    },
    "IDR": {
        "displayName": "Indonesische Rupiah",
        "displayName-count-one": "Indonesische Rupiah",
        "displayName-count-other": "Indonesische Rupiah",
        "symbol": "IDR",
        "symbol-alt-narrow": "Rp"
    },
    "IEP": {
        "displayName": "Irisches Pfund",
        "displayName-count-one": "Irisches Pfund",
        "displayName-count-other": "Irische Pfund",
        "symbol": "IEP"
    },
    "ILP": {
        "displayName": "Israelisches Pfund",
        "displayName-count-one": "Israelisches Pfund",
        "displayName-count-other": "Israelische Pfund",
        "symbol": "ILP"
    },
    "ILR": {
        "displayName": "Israelischer Schekel (1980-1985)",
        "displayName-count-one": "Israelischer Schekel (1980-1985)",
        "displayName-count-other": "Israelische Schekel (1980-1985)"
    },
    "ILS": {
        "displayName": "Israelischer Neuer Schekel",
        "displayName-count-one": "Israelischer Neuer Schekel",
        "displayName-count-other": "Israelische Neue Schekel",
        "symbol": "₪",
        "symbol-alt-narrow": "₪"
    },
    "INR": {
        "displayName": "Indische Rupie",
        "displayName-count-one": "Indische Rupie",
        "displayName-count-other": "Indische Rupien",
        "symbol": "₹",
        "symbol-alt-narrow": "₹"
    },
    "IQD": {
        "displayName": "Irakischer Dinar",
        "displayName-count-one": "Irakischer Dinar",
        "displayName-count-other": "Irakische Dinar",
        "symbol": "IQD"
    },
    "IRR": {

```

```
    "displayName": "Iranischer Rial",
    "displayName-count-one": "Iranischer Rial",
    "displayName-count-other": "Iranische Rial",
    "symbol": "IRR"
  },
  "ISJ": {
    "displayName": "Isländische Krone (1918-1981)",
    "displayName-count-one": "Isländische Krone (1918-1981)",
    "displayName-count-other": "Isländische Kronen (1918-1981)"
  },
  "ISK": {
    "displayName": "Isländische Krone",
    "displayName-count-one": "Isländische Krone",
    "displayName-count-other": "Isländische Kronen",
    "symbol": "ISK",
    "symbol-alt-narrow": "kr"
  },
  "ITL": {
    "displayName": "Italienische Lira",
    "displayName-count-one": "Italienische Lira",
    "displayName-count-other": "Italienische Lire",
    "symbol": "ITL"
  },
  "JMD": {
    "displayName": "Jamaika-Dollar",
    "displayName-count-one": "Jamaika-Dollar",
    "displayName-count-other": "Jamaika-Dollar",
    "symbol": "JMD",
    "symbol-alt-narrow": "$"
  },
  "JOD": {
    "displayName": "Jordanischer Dinar",
    "displayName-count-one": "Jordanischer Dinar",
    "displayName-count-other": "Jordanische Dinar",
    "symbol": "JOD"
  },
  "JPY": {
    "displayName": "Japanischer Yen",
    "displayName-count-one": "Japanischer Yen",
    "displayName-count-other": "Japanische Yen",
    "symbol": "¥",
    "symbol-alt-narrow": "¥"
  },
  "KES": {
    "displayName": "Kenia-Schilling",
    "displayName-count-one": "Kenia-Schilling",
    "displayName-count-other": "Kenia-Schilling",
    "symbol": "KES"
  },
  "KGS": {
    "displayName": "Kirgisischer Som",
    "displayName-count-one": "Kirgisischer Som",
    "displayName-count-other": "Kirgisische Som",
    "symbol": "KGS"
  },
  "KHR": {
    "displayName": "Kambodschanischer Riel",
```

```

        "displayName-count-one": "Kambodschanischer Riel",
        "displayName-count-other": "Kambodschanische Riel",
        "symbol": "KHR",
        "symbol-alt-narrow": "៛"
    },
    "KMF": {
        "displayName": "Komoren-Franc",
        "displayName-count-one": "Komoren-Franc",
        "displayName-count-other": "Komoren-Francs",
        "symbol": "KMF",
        "symbol-alt-narrow": "FC"
    },
    "KPW": {
        "displayName": "Nordkoreanischer Won",
        "displayName-count-one": "Nordkoreanischer Won",
        "displayName-count-other": "Nordkoreanische Won",
        "symbol": "KPW",
        "symbol-alt-narrow": "₩"
    },
    "KRH": {
        "displayName": "Südkoreanischer Hwan (1953-1962)",
        "displayName-count-one": "Südkoreanischer Hwan (1953-1962)",
        "displayName-count-other": "Südkoreanischer Hwan (1953-1962)",
        "symbol": "KRH"
    },
    "KRO": {
        "displayName": "Südkoreanischer Won (1945-1953)",
        "displayName-count-one": "Südkoreanischer Won (1945-1953)",
        "displayName-count-other": "Südkoreanischer Won (1945-1953)",
        "symbol": "KRO"
    },
    "KRW": {
        "displayName": "Südkoreanischer Won",
        "displayName-count-one": "Südkoreanischer Won",
        "displayName-count-other": "Südkoreanische Won",
        "symbol": "₩",
        "symbol-alt-narrow": "₩"
    },
    "KWD": {
        "displayName": "Kuwait-Dinar",
        "displayName-count-one": "Kuwait-Dinar",
        "displayName-count-other": "Kuwait-Dinar",
        "symbol": "KWD"
    },
    "KYD": {
        "displayName": "Kaiman-Dollar",
        "displayName-count-one": "Kaiman-Dollar",
        "displayName-count-other": "Kaiman-Dollar",
        "symbol": "KYD",
        "symbol-alt-narrow": "$"
    },
    "KZT": {
        "displayName": "Kasachischer Tenge",
        "displayName-count-one": "Kasachischer Tenge",
        "displayName-count-other": "Kasachische Tenge",
        "symbol": "KZT",

```



```

        "symbol-alt-narrow": "₭"
    },
    "LAK": {
        "displayName": "Laotischer Kip",
        "displayName-count-one": "Laotischer Kip",
        "displayName-count-other": "Laotische Kip",
        "symbol": "LAK",
        "symbol-alt-narrow": "₭"
    },
    "LBP": {
        "displayName": "Libanesisches Pfund",
        "displayName-count-one": "Libanesisches Pfund",
        "displayName-count-other": "Libanesische Pfund",
        "symbol": "LBP",
        "symbol-alt-narrow": "L£"
    },
    "LKR": {
        "displayName": "Sri-Lanka-Rupie",
        "displayName-count-one": "Sri-Lanka-Rupie",
        "displayName-count-other": "Sri-Lanka-Rupien",
        "symbol": "LKR",
        "symbol-alt-narrow": "Rs"
    },
    "LRD": {
        "displayName": "Liberianischer Dollar",
        "displayName-count-one": "Liberianischer Dollar",
        "displayName-count-other": "Liberianische Dollar",
        "symbol": "LRD",
        "symbol-alt-narrow": "$"
    },
    "LSL": {
        "displayName": "Loti",
        "displayName-count-one": "Loti",
        "displayName-count-other": "Loti",
        "symbol": "LSL"
    },
    "LTL": {
        "displayName": "Litauischer Litas",
        "displayName-count-one": "Litauischer Litas",
        "displayName-count-other": "Litauische Litas",
        "symbol": "LTL",
        "symbol-alt-narrow": "Lt"
    },
    "LTT": {
        "displayName": "Litauischer Talonas",
        "displayName-count-one": "Litauische Talonas",
        "displayName-count-other": "Litauische Talonas",
        "symbol": "LTT"
    },
    "LUC": {
        "displayName": "Luxemburgischer Franc (konvertibel)",
        "displayName-count-one": "Luxemburgische Franc (konvertibel)",
        "displayName-count-other": "Luxemburgische Franc (konvertibel)",
        "symbol": "LUC"
    },
    "LUF": {
        "displayName": "Luxemburgischer Franc",

```

```

        "displayName-count-one": "Luxemburgische Franc",
        "displayName-count-other": "Luxemburgische Franc",
        "symbol": "LUF"
    },
    "LUL": {
        "displayName": "Luxemburgischer Finanz-Franc",
        "displayName-count-one": "Luxemburgische Finanz-Franc",
        "displayName-count-other": "Luxemburgische Finanz-Franc",
        "symbol": "LUL"
    },
    "LVL": {
        "displayName": "Lettischer Lats",
        "displayName-count-one": "Lettischer Lats",
        "displayName-count-other": "Lettische Lats",
        "symbol": "LVL",
        "symbol-alt-narrow": "Ls"
    },
    "LVR": {
        "displayName": "Lettischer Rubel",
        "displayName-count-one": "Lettische Rubel",
        "displayName-count-other": "Lettische Rubel",
        "symbol": "LVR"
    },
    "LYD": {
        "displayName": "Libyscher Dinar",
        "displayName-count-one": "Libyscher Dinar",
        "displayName-count-other": "Libysche Dinar",
        "symbol": "LYD"
    },
    "MAD": {
        "displayName": "Marokkanischer Dirham",
        "displayName-count-one": "Marokkanischer Dirham",
        "displayName-count-other": "Marokkanische Dirham",
        "symbol": "MAD"
    },
    "MAF": {
        "displayName": "Marokkanischer Franc",
        "displayName-count-one": "Marokkanische Franc",
        "displayName-count-other": "Marokkanische Franc",
        "symbol": "MAF"
    },
    "MCF": {
        "displayName": "Monegassischer Franc",
        "displayName-count-one": "Monegassischer Franc",
        "displayName-count-other": "Monegassische Franc",
        "symbol": "MCF"
    },
    "MDC": {
        "displayName": "Moldau-Cupon",
        "displayName-count-one": "Moldau-Cupon",
        "displayName-count-other": "Moldau-Cupon",
        "symbol": "MDC"
    },
    "MDL": {
        "displayName": "Moldau-Leu",
        "displayName-count-one": "Moldau-Leu",
        "displayName-count-other": "Moldau-Leu",

```

```

        "symbol": "MDL"
    },
    "MGA": {
        "displayName": "Madagaskar-Ariary",
        "displayName-count-one": "Madagaskar-Ariary",
        "displayName-count-other": "Madagaskar-Ariary",
        "symbol": "MGA",
        "symbol-alt-narrow": "Ar"
    },
    "MGF": {
        "displayName": "Madagaskar-Franc",
        "displayName-count-one": "Madagaskar-Franc",
        "displayName-count-other": "Madagaskar-Franc",
        "symbol": "MGF"
    },
    "MKD": {
        "displayName": "Mazedonischer Denar",
        "displayName-count-one": "Mazedonischer Denar",
        "displayName-count-other": "Mazedonische Denari",
        "symbol": "MKD"
    },
    "MKN": {
        "displayName": "Mazedonischer Denar (1992-1993)",
        "displayName-count-one": "Mazedonischer Denar (1992-1993)",
        "displayName-count-other": "Mazedonische Denar (1992-1993)",
        "symbol": "MKN"
    },
    "MLF": {
        "displayName": "Malischer Franc",
        "displayName-count-one": "Malische Franc",
        "displayName-count-other": "Malische Franc",
        "symbol": "MLF"
    },
    "MMK": {
        "displayName": "Myanmarischer Kyat",
        "displayName-count-one": "Myanmarischer Kyat",
        "displayName-count-other": "Myanmarische Kyat",
        "symbol": "MMK",
        "symbol-alt-narrow": "K"
    },
    "MNT": {
        "displayName": "Mongolischer Tögrög",
        "displayName-count-one": "Mongolischer Tögrög",
        "displayName-count-other": "Mongolische Tögrög",
        "symbol": "MNT",
        "symbol-alt-narrow": "₮"
    },
    "MOP": {
        "displayName": "Macao-Pataca",
        "displayName-count-one": "Macao-Pataca",
        "displayName-count-other": "Macao-Pataca",
        "symbol": "MOP"
    },
    "MRO": {
        "displayName": "Mauretanischer Ouguiya",
        "displayName-count-one": "Mauretanischer Ouguiya",
        "displayName-count-other": "Mauretanische Ouguiya",

```

```

        "symbol": "MRO"
    },
    "MTL": {
        "displayName": "Maltesische Lira",
        "displayName-count-one": "Maltesische Lira",
        "displayName-count-other": "Maltesische Lira",
        "symbol": "MTL"
    },
    "MTP": {
        "displayName": "Maltesisches Pfund",
        "displayName-count-one": "Maltesische Pfund",
        "displayName-count-other": "Maltesische Pfund",
        "symbol": "MTP"
    },
    "MUR": {
        "displayName": "Mauritius-Rupie",
        "displayName-count-one": "Mauritius-Rupie",
        "displayName-count-other": "Mauritius-Rupien",
        "symbol": "MUR",
        "symbol-alt-narrow": "Rs"
    },
    "MVP": {
        "displayName": "Malediven-Rupie (alt)",
        "displayName-count-one": "Malediven-Rupie (alt)",
        "displayName-count-other": "Malediven-Rupien (alt)"
    },
    "MVR": {
        "displayName": "Malediven-Rufiyaa",
        "displayName-count-one": "Malediven-Rufiyaa",
        "displayName-count-other": "Malediven-Rupien",
        "symbol": "MVR"
    },
    "MWK": {
        "displayName": "Malawi-Kwacha",
        "displayName-count-one": "Malawi-Kwacha",
        "displayName-count-other": "Malawi-Kwacha",
        "symbol": "MWK"
    },
    "MXN": {
        "displayName": "Mexikanischer Peso",
        "displayName-count-one": "Mexikanischer Peso",
        "displayName-count-other": "Mexikanische Pesos",
        "symbol": "MX$",
        "symbol-alt-narrow": "$"
    },
    "MXP": {
        "displayName": "Mexikanischer Silber-Peso (1861-1992)",
        "displayName-count-one": "Mexikanische Silber-Peso (1861-1992)",
        "displayName-count-other": "Mexikanische Silber-Pesos (1861-
1992)",
        "symbol": "MXP"
    },
    "MXV": {
        "displayName": "Mexicanischer Unidad de Inversion (UDI)",
        "displayName-count-one": "Mexicanischer Unidad de Inversion
(UDI)",

```

```

        "displayName-count-other": "Mexikanische Unidad de Inversion
(UDI) ",
        "symbol": "MXV"
    },
    "MYR": {
        "displayName": "Malaysischer Ringgit",
        "displayName-count-one": "Malaysischer Ringgit",
        "displayName-count-other": "Malaysische Ringgit",
        "symbol": "MYR",
        "symbol-alt-narrow": "RM"
    },
    "MZE": {
        "displayName": "Mosambikanischer Escudo",
        "displayName-count-one": "Mozambikanische Escudo",
        "displayName-count-other": "Mozambikanische Escudo",
        "symbol": "MZE"
    },
    "MZM": {
        "displayName": "Mosambikanischer Metical (1980-2006)",
        "displayName-count-one": "Mosambikanischer Metical (1980-2006)",
        "displayName-count-other": "Mosambikanische Meticais (1980-
2006) ",
        "symbol": "MZM"
    },
    "MZN": {
        "displayName": "Mosambikanischer Metical",
        "displayName-count-one": "Mosambikanischer Metical",
        "displayName-count-other": "Mosambikanische Meticais",
        "symbol": "MZN"
    },
    "NAD": {
        "displayName": "Namibia-Dollar",
        "displayName-count-one": "Namibia-Dollar",
        "displayName-count-other": "Namibia-Dollar",
        "symbol": "NAD",
        "symbol-alt-narrow": "$"
    },
    "NGN": {
        "displayName": "Nigerianischer Naira",
        "displayName-count-one": "Nigerianischer Naira",
        "displayName-count-other": "Nigerianische Naira",
        "symbol": "NGN",
        "symbol-alt-narrow": "₦"
    },
    "NIC": {
        "displayName": "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-one": "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-other": "Nicaraguanische Córdoba (1988-1991)",
        "symbol": "NIC"
    },
    "NIO": {
        "displayName": "Nicaragua-Córdoba",
        "displayName-count-one": "Nicaragua-Córdoba",
        "displayName-count-other": "Nicaragua-Córdobas",
        "symbol": "NIO",
        "symbol-alt-narrow": "C$"
    },
    },

```

```
"NLG": {
  "displayName": "Niederländischer Gulden",
  "displayName-count-one": "Niederländischer Gulden",
  "displayName-count-other": "Niederländische Gulden",
  "symbol": "NLG"
},
"NOK": {
  "displayName": "Norwegische Krone",
  "displayName-count-one": "Norwegische Krone",
  "displayName-count-other": "Norwegische Kronen",
  "symbol": "NOK",
  "symbol-alt-narrow": "kr"
},
"NPR": {
  "displayName": "Nepalesische Rupie",
  "displayName-count-one": "Nepalesische Rupie",
  "displayName-count-other": "Nepalesische Rupien",
  "symbol": "NPR",
  "symbol-alt-narrow": "Rs"
},
"NZD": {
  "displayName": "Neuseeland-Dollar",
  "displayName-count-one": "Neuseeland-Dollar",
  "displayName-count-other": "Neuseeland-Dollar",
  "symbol": "NZ$",
  "symbol-alt-narrow": "$"
},
"OMR": {
  "displayName": "Omanischer Rial",
  "displayName-count-one": "Omanischer Rial",
  "displayName-count-other": "Omanische Rials",
  "symbol": "OMR"
},
"PAB": {
  "displayName": "Panamaischer Balboa",
  "displayName-count-one": "Panamaischer Balboa",
  "displayName-count-other": "Panamaische Balboas",
  "symbol": "PAB"
},
"PEI": {
  "displayName": "Peruanischer Inti",
  "displayName-count-one": "Peruanische Inti",
  "displayName-count-other": "Peruanische Inti",
  "symbol": "PEI"
},
"PEN": {
  "displayName": "Peruanischer Sol",
  "displayName-count-one": "Peruanischer Sol",
  "displayName-count-other": "Peruanische Sol",
  "symbol": "PEN"
},
"PES": {
  "displayName": "Peruanischer Sol (1863-1965)",
  "displayName-count-one": "Peruanischer Sol (1863-1965)",
  "displayName-count-other": "Peruanische Sol (1863-1965)",
  "symbol": "PES"
},
}
```

```
"PGK": {
  "displayName": "Papua-Neuguineischer Kina",
  "displayName-count-one": "Papua-Neuguineischer Kina",
  "displayName-count-other": "Papua-Neuguineische Kina",
  "symbol": "PGK"
},
"PHP": {
  "displayName": "Philippinischer Peso",
  "displayName-count-one": "Philippinischer Peso",
  "displayName-count-other": "Philippinische Pesos",
  "symbol": "PHP",
  "symbol-alt-narrow": "₱"
},
"PKR": {
  "displayName": "Pakistanische Rupie",
  "displayName-count-one": "Pakistanische Rupie",
  "displayName-count-other": "Pakistanische Rupien",
  "symbol": "PKR",
  "symbol-alt-narrow": "Rs"
},
"PLN": {
  "displayName": "Polnischer Złoty",
  "displayName-count-one": "Polnischer Złoty",
  "displayName-count-other": "Polnische Złoty",
  "symbol": "PLN",
  "symbol-alt-narrow": "zł"
},
"PLZ": {
  "displayName": "Polnischer Zloty (1950-1995)",
  "displayName-count-one": "Polnischer Zloty (1950-1995)",
  "displayName-count-other": "Polnische Zloty (1950-1995)",
  "symbol": "PLZ"
},
"PTE": {
  "displayName": "Portugiesischer Escudo",
  "displayName-count-one": "Portugiesische Escudo",
  "displayName-count-other": "Portugiesische Escudo",
  "symbol": "PTE"
},
"PYG": {
  "displayName": "Paraguayischer Guaraní",
  "displayName-count-one": "Paraguayischer Guaraní",
  "displayName-count-other": "Paraguayische Guaraníes",
  "symbol": "PYG",
  "symbol-alt-narrow": "₲"
},
"QAR": {
  "displayName": "Katar-Riyal",
  "displayName-count-one": "Katar-Riyal",
  "displayName-count-other": "Katar-Riyal",
  "symbol": "QAR"
},
"RHD": {
  "displayName": "Rhodesischer Dollar",
  "displayName-count-one": "Rhodesische Dollar",
  "displayName-count-other": "Rhodesische Dollar",
  "symbol": "RHD"
```

```

    },
    "ROL": {
      "displayName": "Rumänischer Leu (1952-2006)",
      "displayName-count-one": "Rumänischer Leu (1952-2006)",
      "displayName-count-other": "Rumänische Leu (1952-2006)",
      "symbol": "ROL"
    },
    "RON": {
      "displayName": "Rumänischer Leu",
      "displayName-count-one": "Rumänischer Leu",
      "displayName-count-other": "Rumänische Leu",
      "symbol": "RON",
      "symbol-alt-narrow": "L"
    },
    "RSD": {
      "displayName": "Serbischer Dinar",
      "displayName-count-one": "Serbischer Dinar",
      "displayName-count-other": "Serbische Dinaren",
      "symbol": "RSD"
    },
    "RUB": {
      "displayName": "Russischer Rubel",
      "displayName-count-one": "Russischer Rubel",
      "displayName-count-other": "Russische Rubel",
      "symbol": "RUB",
      "symbol-alt-narrow": "₽"
    },
    "RUR": {
      "displayName": "Russischer Rubel (1991-1998)",
      "displayName-count-one": "Russischer Rubel (1991-1998)",
      "displayName-count-other": "Russische Rubel (1991-1998)",
      "symbol": "RUR",
      "symbol-alt-narrow": "p."
    },
    "RWF": {
      "displayName": "Ruanda-Franc",
      "displayName-count-one": "Ruanda-Franc",
      "displayName-count-other": "Ruanda-Francis",
      "symbol": "RWF",
      "symbol-alt-narrow": "F.Rw"
    },
    "SAR": {
      "displayName": "Saudi-Rial",
      "displayName-count-one": "Saudi-Rial",
      "displayName-count-other": "Saudi-Rial",
      "symbol": "SAR"
    },
    "SBD": {
      "displayName": "Salomonen-Dollar",
      "displayName-count-one": "Salomonen-Dollar",
      "displayName-count-other": "Salomonen-Dollar",
      "symbol": "SBD",
      "symbol-alt-narrow": "$"
    },
    "SCR": {
      "displayName": "Seychellen-Rupie",
      "displayName-count-one": "Seychellen-Rupie",

```



```

        "displayName-count-other": "Seychellen-Rupien",
        "symbol": "SCR"
    },
    "SDD": {
        "displayName": "Sudanesischer Dinar (1992-2007)",
        "displayName-count-one": "Sudanesischer Dinar (1992-2007)",
        "displayName-count-other": "Sudanesische Dinar (1992-2007)",
        "symbol": "SDD"
    },
    "SDG": {
        "displayName": "Sudanesisches Pfund",
        "displayName-count-one": "Sudanesisches Pfund",
        "displayName-count-other": "Sudanesische Pfund",
        "symbol": "SDG"
    },
    "SDP": {
        "displayName": "Sudanesisches Pfund (1957-1998)",
        "displayName-count-one": "Sudanesisches Pfund (1957-1998)",
        "displayName-count-other": "Sudanesische Pfund (1957-1998)",
        "symbol": "SDP"
    },
    "SEK": {
        "displayName": "Schwedische Krone",
        "displayName-count-one": "Schwedische Krone",
        "displayName-count-other": "Schwedische Kronen",
        "symbol": "SEK",
        "symbol-alt-narrow": "kr"
    },
    "SGD": {
        "displayName": "Singapur-Dollar",
        "displayName-count-one": "Singapur-Dollar",
        "displayName-count-other": "Singapur-Dollar",
        "symbol": "SGD",
        "symbol-alt-narrow": "$"
    },
    "SHP": {
        "displayName": "St. Helena-Pfund",
        "displayName-count-one": "St. Helena-Pfund",
        "displayName-count-other": "St. Helena-Pfund",
        "symbol": "SHP",
        "symbol-alt-narrow": "£"
    },
    "SIT": {
        "displayName": "Slowenischer Tolar",
        "displayName-count-one": "Slowenischer Tolar",
        "displayName-count-other": "Slowenische Tolar",
        "symbol": "SIT"
    },
    "SKK": {
        "displayName": "Slowakische Krone",
        "displayName-count-one": "Slowakische Kronen",
        "displayName-count-other": "Slowakische Kronen",
        "symbol": "SKK"
    },
    "SLL": {
        "displayName": "Sierra-leonischer Leone",
        "displayName-count-one": "Sierra-leonischer Leone",

```

```

        "displayName-count-other": "Sierra-leonische Leones",
        "symbol": "SLL"
    },
    "SOS": {
        "displayName": "Somalia-Schilling",
        "displayName-count-one": "Somalia-Schilling",
        "displayName-count-other": "Somalia-Schilling",
        "symbol": "SOS"
    },
    "SRD": {
        "displayName": "Suriname-Dollar",
        "displayName-count-one": "Suriname-Dollar",
        "displayName-count-other": "Suriname-Dollar",
        "symbol": "SRD",
        "symbol-alt-narrow": "$"
    },
    "SRG": {
        "displayName": "Suriname Gulden",
        "displayName-count-one": "Suriname-Gulden",
        "displayName-count-other": "Suriname-Gulden",
        "symbol": "SRG"
    },
    "SSP": {
        "displayName": "Südsudanesisches Pfund",
        "displayName-count-one": "Südsudanesisches Pfund",
        "displayName-count-other": "Südsudanesische Pfund",
        "symbol": "SSP",
        "symbol-alt-narrow": "£"
    },
    "STD": {
        "displayName": "São-toméischer Dobra",
        "displayName-count-one": "São-toméischer Dobra",
        "displayName-count-other": "São-toméische Dobra",
        "symbol": "STD",
        "symbol-alt-narrow": "Db"
    },
    "SUR": {
        "displayName": "Sowjetischer Rubel",
        "displayName-count-one": "Sowjetische Rubel",
        "displayName-count-other": "Sowjetische Rubel",
        "symbol": "SUR"
    },
    "SVC": {
        "displayName": "El Salvador Colon",
        "displayName-count-one": "El Salvador-Colon",
        "displayName-count-other": "El Salvador-Colon",
        "symbol": "SVC"
    },
    "SYP": {
        "displayName": "Syrisches Pfund",
        "displayName-count-one": "Syrisches Pfund",
        "displayName-count-other": "Syrische Pfund",
        "symbol": "SYP",
        "symbol-alt-narrow": "SYP"
    },
    "SZL": {
        "displayName": "Swasiländischer Lilangeni",

```

```

        "displayName-count-one": "Swasiländischer Lilangeni",
        "displayName-count-other": "Swasiländische Emalangeni",
        "symbol": "SZL"
    },
    "THB": {
        "displayName": "Thailändischer Baht",
        "displayName-count-one": "Thailändischer Baht",
        "displayName-count-other": "Thailändische Baht",
        "symbol": "฿",
        "symbol-alt-narrow": "฿"
    },
    "TJR": {
        "displayName": "Tadschikistan Rubel",
        "displayName-count-one": "Tadschikistan-Rubel",
        "displayName-count-other": "Tadschikistan-Rubel",
        "symbol": "TJR"
    },
    "TJS": {
        "displayName": "Tadschikistan-Somoni",
        "displayName-count-one": "Tadschikistan-Somoni",
        "displayName-count-other": "Tadschikistan-Somoni",
        "symbol": "TJS"
    },
    "TMM": {
        "displayName": "Turkmenistan-Manat (1993-2009)",
        "displayName-count-one": "Turkmenistan-Manat (1993-2009)",
        "displayName-count-other": "Turkmenistan-Manat (1993-2009)",
        "symbol": "TMM"
    },
    "TMT": {
        "displayName": "Turkmenistan-Manat",
        "displayName-count-one": "Turkmenistan-Manat",
        "displayName-count-other": "Turkmenistan-Manat",
        "symbol": "TMT"
    },
    "TND": {
        "displayName": "Tunesischer Dinar",
        "displayName-count-one": "Tunesischer Dinar",
        "displayName-count-other": "Tunesische Dinar",
        "symbol": "TND"
    },
    "TOP": {
        "displayName": "Tongaischer Paʻanga",
        "displayName-count-one": "Tongaischer Paʻanga",
        "displayName-count-other": "Tongaische Paʻanga",
        "symbol": "TOP",
        "symbol-alt-narrow": "T$"
    },
    "TPE": {
        "displayName": "Timor-Escudo",
        "displayName-count-one": "Timor-Escudo",
        "displayName-count-other": "Timor-Escudo",
        "symbol": "TPE"
    },
    "TRL": {
        "displayName": "Türkische Lira (1922-2005)",
        "displayName-count-one": "Türkische Lira (1922-2005)",

```

```

        "displayName-count-other": "Türkische Lira (1922-2005)",
        "symbol": "TRL"
    },
    "TRY": {
        "displayName": "Türkische Lira",
        "displayName-count-one": "Türkische Lira",
        "displayName-count-other": "Türkische Lira",
        "symbol": "TRY",
        "symbol-alt-narrow": "₺",
        "symbol-alt-variant": "TL"
    },
    "TTD": {
        "displayName": "Trinidad und Tobago-Dollar",
        "displayName-count-one": "Trinidad und Tobago-Dollar",
        "displayName-count-other": "Trinidad und Tobago-Dollar",
        "symbol": "TTD",
        "symbol-alt-narrow": "$"
    },
    "TWD": {
        "displayName": "Neuer Taiwan-Dollar",
        "displayName-count-one": "Neuer Taiwan-Dollar",
        "displayName-count-other": "Neue Taiwan-Dollar",
        "symbol": "NT$",
        "symbol-alt-narrow": "NT$"
    },
    "TZS": {
        "displayName": "Tansania-Schilling",
        "displayName-count-one": "Tansania-Schilling",
        "displayName-count-other": "Tansania-Schilling",
        "symbol": "TZS"
    },
    "UAH": {
        "displayName": "Ukrainische Hrywnja",
        "displayName-count-one": "Ukrainische Hrywnja",
        "displayName-count-other": "Ukrainische Hrywen",
        "symbol": "UAH",
        "symbol-alt-narrow": "₴"
    },
    "UAK": {
        "displayName": "Ukrainischer Karbovanetz",
        "displayName-count-one": "Ukrainische Karbovanetz",
        "displayName-count-other": "Ukrainische Karbovanetz",
        "symbol": "UAK"
    },
    "UGS": {
        "displayName": "Uganda-Schilling (1966-1987)",
        "displayName-count-one": "Uganda-Schilling (1966-1987)",
        "displayName-count-other": "Uganda-Schilling (1966-1987)",
        "symbol": "UGS"
    },
    "UGX": {
        "displayName": "Uganda-Schilling",
        "displayName-count-one": "Uganda-Schilling",
        "displayName-count-other": "Uganda-Schilling",
        "symbol": "UGX"
    },
    "USD": {

```

```

        "displayName": "US-Dollar",
        "displayName-count-one": "US-Dollar",
        "displayName-count-other": "US-Dollar",
        "symbol": "$",
        "symbol-alt-narrow": "$"
    },
    "USN": {
        "displayName": "US Dollar (Nächster Tag)",
        "displayName-count-one": "US-Dollar (Nächster Tag)",
        "displayName-count-other": "US-Dollar (Nächster Tag)",
        "symbol": "USN"
    },
    "USS": {
        "displayName": "US Dollar (Gleicher Tag)",
        "displayName-count-one": "US-Dollar (Gleicher Tag)",
        "displayName-count-other": "US-Dollar (Gleicher Tag)",
        "symbol": "USS"
    },
    "UYI": {
        "displayName": "Uruguayischer Peso (Indexierte
Rechnungseinheiten)",
        "displayName-count-one": "Uruguayischer Peso (Indexierte
Rechnungseinheiten)",
        "displayName-count-other": "Uruguayische Pesos (Indexierte
Rechnungseinheiten)",
        "symbol": "UYI"
    },
    "UYP": {
        "displayName": "Uruguayischer Peso (1975-1993)",
        "displayName-count-one": "Uruguayischer Peso (1975-1993)",
        "displayName-count-other": "Uruguayische Pesos (1975-1993)",
        "symbol": "UYP"
    },
    "UYU": {
        "displayName": "Uruguayischer Peso",
        "displayName-count-one": "Uruguayischer Peso",
        "displayName-count-other": "Uruguayische Pesos",
        "symbol": "UYU",
        "symbol-alt-narrow": "$"
    },
    "UZS": {
        "displayName": "Usbekistan-Sum",
        "displayName-count-one": "Usbekistan-Sum",
        "displayName-count-other": "Usbekistan-Sum",
        "symbol": "UZS"
    },
    "VEB": {
        "displayName": "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-one": "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-other": "Venezolanische Bolíva
2008)",
        "symbol": "VEB"
    },
    "VEF": {
        "displayName": "Venezolanischer Bolívar",
        "displayName-count-one": "Venezolanischer Bolívar",
        "displayName-count-other": "Venezolanische Bolíva

```

```

        "symbol": "VEF",
        "symbol-alt-narrow": "Bs"
    },
    "VND": {
        "displayName": "Vietnamesischer Dong",
        "displayName-count-one": "Vietnamesischer Dong",
        "displayName-count-other": "Vietnamesische Dong",
        "symbol": "₫",
        "symbol-alt-narrow": "₫"
    },
    "VNN": {
        "displayName": "Vietnamesischer Dong (1978-1985)",
        "displayName-count-one": "Vietnamesischer Dong (1978-1985)",
        "displayName-count-other": "Vietnamesische Dong (1978-1985)",
        "symbol": "VNN"
    },
    "VUV": {
        "displayName": "Vanuatu-Vatu",
        "displayName-count-one": "Vanuatu-Vatu",
        "displayName-count-other": "Vanuatu-Vatu",
        "symbol": "VUV"
    },
    "WST": {
        "displayName": "Samoanischer Tala",
        "displayName-count-one": "Samoanischer Tala",
        "displayName-count-other": "Samoanische Tala",
        "symbol": "WST"
    },
    "XAF": {
        "displayName": "CFA-Franc (BEAC)",
        "displayName-count-one": "CFA-Franc (BEAC)",
        "displayName-count-other": "CFA-Franc (BEAC)",
        "symbol": "FCFA"
    },
    "XAG": {
        "displayName": "Unze Silber",
        "displayName-count-one": "Unze Silber",
        "displayName-count-other": "Unzen Silber",
        "symbol": "XAG"
    },
    "XAU": {
        "displayName": "Unze Gold",
        "displayName-count-one": "Unze Gold",
        "displayName-count-other": "Unzen Gold",
        "symbol": "XAU"
    },
    "XBA": {
        "displayName": "Europäische Rechnungseinheit",
        "displayName-count-one": "Europäische Rechnungseinheiten",
        "displayName-count-other": "Europäische Rechnungseinheiten",
        "symbol": "XBA"
    },
    "XBB": {
        "displayName": "Europäische Währungseinheit (XBB)",
        "displayName-count-one": "Europäische Währungseinheiten (XBB)",
        "displayName-count-other": "Europäische Währungseinheiten (XBB)",
        "symbol": "XBB"
    }

```

```

    },
    "XBC": {
      "displayName": "Europäische Rechnungseinheit (XBC)",
      "displayName-count-one": "Europäische Rechnungseinheiten (XBC)",
      "displayName-count-other": "Europäische Rechnungseinheiten
(XBC) ",
      "symbol": "XBC"
    },
    "XBD": {
      "displayName": "Europäische Rechnungseinheit (XBD)",
      "displayName-count-one": "Europäische Rechnungseinheiten (XBD)",
      "displayName-count-other": "Europäische Rechnungseinheiten
(XBD) ",
      "symbol": "XBD"
    },
    "XCD": {
      "displayName": "Ostkaribischer Dollar",
      "displayName-count-one": "Ostkaribischer Dollar",
      "displayName-count-other": "Ostkaribische Dollar",
      "symbol": "EC$",
      "symbol-alt-narrow": "$"
    },
    "XDR": {
      "displayName": "Sonderziehungsrechte",
      "displayName-count-one": "Sonderziehungsrechte",
      "displayName-count-other": "Sonderziehungsrechte",
      "symbol": "XDR"
    },
    "XEU": {
      "displayName": "Europäische Währungseinheit (XEU)",
      "displayName-count-one": "Europäische Währungseinheiten (XEU)",
      "displayName-count-other": "Europäische Währungseinheiten (XEU)",
      "symbol": "XEU"
    },
    "XFO": {
      "displayName": "Französischer Gold-Franc",
      "displayName-count-one": "Französische Gold-Franc",
      "displayName-count-other": "Französische Gold-Franc",
      "symbol": "XFO"
    },
    "XFU": {
      "displayName": "Französischer UIC-Franc",
      "displayName-count-one": "Französische UIC-Franc",
      "displayName-count-other": "Französische UIC-Franc",
      "symbol": "XFU"
    },
    "XOF": {
      "displayName": "CFA-Franc (BCEAO) ",
      "displayName-count-one": "CFA-Franc (BCEAO) ",
      "displayName-count-other": "CFA-Francs (BCEAO) ",
      "symbol": "CFA"
    },
    "XPD": {
      "displayName": "Unze Palladium",
      "displayName-count-one": "Unze Palladium",
      "displayName-count-other": "Unzen Palladium",
      "symbol": "XPD"
    }
  }

```

```

    },
    "XPF": {
      "displayName": "CFP-Franc",
      "displayName-count-one": "CFP-Franc",
      "displayName-count-other": "CFP-Franc",
      "symbol": "CFPF"
    },
    "XPT": {
      "displayName": "Unze Platin",
      "displayName-count-one": "Unze Platin",
      "displayName-count-other": "Unzen Platin",
      "symbol": "XPT"
    },
    "XRE": {
      "displayName": "RINET Funds",
      "displayName-count-one": "RINET Funds",
      "displayName-count-other": "RINET Funds",
      "symbol": "XRE"
    },
    "XSU": {
      "displayName": "SUCRE",
      "displayName-count-one": "SUCRE",
      "displayName-count-other": "SUCRE",
      "symbol": "XSU"
    },
    "XTS": {
      "displayName": "Testwährung",
      "displayName-count-one": "Testwährung",
      "displayName-count-other": "Testwährung",
      "symbol": "XTS"
    },
    "XUA": {
      "displayName": "Rechnungseinheit der AfEB",
      "displayName-count-one": "Rechnungseinheit der AfEB",
      "displayName-count-other": "Rechnungseinheiten der AfEB",
      "symbol": "XUA"
    },
    "XXX": {
      "displayName": "Unbekannte Währung",
      "displayName-count-one": "(unbekannte Währung)",
      "displayName-count-other": "(unbekannte Währung)",
      "symbol": "XXX"
    },
    "YDD": {
      "displayName": "Jemen-Dinar",
      "displayName-count-one": "Jemen-Dinar",
      "displayName-count-other": "Jemen-Dinar",
      "symbol": "YDD"
    },
    "YER": {
      "displayName": "Jemen-Rial",
      "displayName-count-one": "Jemen-Rial",
      "displayName-count-other": "Jemen-Rial",
      "symbol": "YER"
    },
    "YUD": {
      "displayName": "Jugoslawischer Dinar (1966-1990)",

```



```

        "displayName-count-one": "Jugoslawischer Dinar (1966-1990)",
        "displayName-count-other": "Jugoslawische Dinar (1966-1990)",
        "symbol": "YUD"
    },
    "YUM": {
        "displayName": "Jugoslawischer Neuer Dinar (1994-2002)",
        "displayName-count-one": "Jugoslawischer Neuer Dinar (1994-
2002) ",
        "displayName-count-other": "Jugoslawische Neue Dinar (1994-
2002) ",
        "symbol": "YUM"
    },
    "YUN": {
        "displayName": "Jugoslawischer Dinar (konvertibel)",
        "displayName-count-one": "Jugoslawische Dinar (konvertibel)",
        "displayName-count-other": "Jugoslawische Dinar (konvertibel)",
        "symbol": "YUN"
    },
    "YUR": {
        "displayName": "Jugoslawischer reformierter Dinar (1992-1993)",
        "displayName-count-one": "Jugoslawischer reformierter Dinar
(1992-1993) ",
        "displayName-count-other": "Jugoslawische reformierte Dinar
(1992-1993) ",
        "symbol": "YUR"
    },
    "ZAL": {
        "displayName": "Südafrikanischer Rand (Finanz)",
        "displayName-count-one": "Südafrikanischer Rand (Finanz)",
        "displayName-count-other": "Südafrikanischer Rand (Finanz)",
        "symbol": "ZAL"
    },
    "ZAR": {
        "displayName": "Südafrikanischer Rand",
        "displayName-count-one": "Südafrikanischer Rand",
        "displayName-count-other": "Südafrikanische Rand",
        "symbol": "ZAR",
        "symbol-alt-narrow": "R"
    },
    "ZMK": {
        "displayName": "Kwacha (1968-2012)",
        "displayName-count-one": "Kwacha (1968-2012)",
        "displayName-count-other": "Kwacha (1968-2012)",
        "symbol": "ZMK"
    },
    "ZMW": {
        "displayName": "Kwacha",
        "displayName-count-one": "Kwacha",
        "displayName-count-other": "Kwacha",
        "symbol": "ZMW",
        "symbol-alt-narrow": "K"
    },
    "ZRN": {
        "displayName": "Zaire-Neuer Zaïre (1993-1998)",
        "displayName-count-one": "Zaire-Neuer Zaïre (1993-1998)",
        "displayName-count-other": "Zaire-Neue Zaïre (1993-1998)",
        "symbol": "ZRN"
    }

```

```

    },
    "ZRZ": {
      "displayName": "Zaire-Zaire (1971-1993)",
      "displayName-count-one": "Zaire-Zaire (1971-1993)",
      "displayName-count-other": "Zaire-Zaire (1971-1993)",
      "symbol": "ZRZ"
    },
    "ZWD": {
      "displayName": "Simbabwe-Dollar (1980-2008)",
      "displayName-count-one": "Simbabwe-Dollar (1980-2008)",
      "displayName-count-other": "Simbabwe-Dollar (1980-2008)",
      "symbol": "ZWD"
    },
    "ZWL": {
      "displayName": "Simbabwe-Dollar (2009)",
      "displayName-count-one": "Simbabwe-Dollar (2009)",
      "displayName-count-other": "Simbabwe-Dollar (2009)",
      "symbol": "ZWL"
    },
    "ZWR": {
      "displayName": "Simbabwe-Dollar (2008)",
      "displayName-count-one": "Simbabwe-Dollar (2008)",
      "displayName-count-other": "Simbabwe-Dollar (2008)",
      "symbol": "ZWR"
    }
  },
},
},
},
},
}

```

CURRENCIES.JSX

```

{
  "main";
  {
    "de";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 13259 $",
          "_cldrVersion";
          "31";
        }
        "language";
        "de";
      }
      "numbers";
      {
        "currencies";
        {
          "ADP";
          {

```

```

        "displayName";
        "Andorranische Pesete",
        "displayName-count-one";
        "Andorranische Pesete",
        "displayName-count-other";
        "Andorranische Peseten",
        "symbol";
        "ADP";
    }
    "AED";
    {
        "displayName";
        "VAE-Dirham",
        "displayName-count-one";
        "VAE-Dirham",
        "displayName-count-other";
        "VAE-Dirham",
        "symbol";
        "AED";
    }
    "AFA";
    {
        "displayName";
        "Afghanische Afghani (1927-2002)",
        "displayName-count-one";
        "Afghanische Afghani (1927-2002)",
        "displayName-count-other";
        "Afghanische Afghani (1927-2002)",
        "symbol";
        "AFA";
    }
    "AFN";
    {
        "displayName";
        "Afghanischer Afghani",
        "displayName-count-one";
        "Afghanischer Afghani",
        "displayName-count-other";
        "Afghanische Afghani",
        "symbol";
        "AFN";
    }
    "ALK";
    {
        "displayName";
        "Albanischer Lek (1946-1965)",
        "displayName-count-one";
        "Albanischer Lek (1946-1965)",
        "displayName-count-other";
        "Albanische Lek (1946-1965)";
    }
    "ALL";
    {
        "displayName";
        "Albanischer Lek",
        "displayName-count-one";
        "Albanischer Lek",

```

```

        "displayName-count-other";
        "Albanische Lek",
        "symbol";
        "ALL";
    }
    "AMD";
    {
        "displayName";
        "Armenischer Dram",
        "displayName-count-one";
        "Armenischer Dram",
        "displayName-count-other";
        "Armenische Dram",
        "symbol";
        "AMD";
    }
    "ANG";
    {
        "displayName";
        "Niederländische-Antillen-Gulden",
        "displayName-count-one";
        "Niederländische-Antillen-Gulden",
        "displayName-count-other";
        "Niederländische-Antillen-Gulden",
        "symbol";
        "ANG";
    }
    "AOA";
    {
        "displayName";
        "Angolanischer Kwanza",
        "displayName-count-one";
        "Angolanischer Kwanza",
        "displayName-count-other";
        "Angolanische Kwanza",
        "symbol";
        "AOA",
        "symbol-alt-narrow";
        "Kz";
    }
    "AOK";
    {
        "displayName";
        "Angolanischer Kwanza (1977-1990)",
        "displayName-count-one";
        "Angolanischer Kwanza (1977-1990)",
        "displayName-count-other";
        "Angolanische Kwanza (1977-1990)",
        "symbol";
        "AOK";
    }
    "AON";
    {
        "displayName";
        "Angolanischer Neuer Kwanza (1990-2000)",
        "displayName-count-one";
        "Angolanischer Neuer Kwanza (1990-2000)",

```

```

        "displayName-count-other";
        "Angolanische Neue Kwanza (1990-2000)",
        "symbol";
        "AON";
    }
    "AOR";
    {
        "displayName";
        "Angolanischer Kwanza Reajustado (1995-1999)",
        "displayName-count-one";
        "Angolanischer Kwanza Reajustado (1995-1999)",
        "displayName-count-other";
        "Angolanische Kwanza Reajustado (1995-1999)",
        "symbol";
        "AOR";
    }
    "ARA";
    {
        "displayName";
        "Argentinischer Austral",
        "displayName-count-one";
        "Argentinischer Austral",
        "displayName-count-other";
        "Argentinische Austral",
        "symbol";
        "ARA";
    }
    "ARL";
    {
        "displayName";
        "Argentinischer Peso Ley (1970-1983)",
        "displayName-count-one";
        "Argentinischer Peso Ley (1970-1983)",
        "displayName-count-other";
        "Argentinische Pesos Ley (1970-1983)",
        "symbol";
        "ARL";
    }
    "ARM";
    {
        "displayName";
        "Argentinischer Peso (1881-1970)",
        "displayName-count-one";
        "Argentinischer Peso (1881-1970)",
        "displayName-count-other";
        "Argentinische Pesos (1881-1970)",
        "symbol";
        "ARM";
    }
    "ARP";
    {
        "displayName";
        "Argentinischer Peso (1983-1985)",
        "displayName-count-one";
        "Argentinischer Peso (1983-1985)",
        "displayName-count-other";
        "Argentinische Peso (1983-1985)",

```

```

        "symbol";
        "ARP";
    }
    "ARS";
    {
        "displayName";
        "Argentinischer Peso",
        "displayName-count-one";
        "Argentinischer Peso",
        "displayName-count-other";
        "Argentinische Pesos",
        "symbol";
        "ARS",
        "symbol-alt-narrow";
        "$";
    }
    "ATS";
    {
        "displayName";
        "Österreichischer Schilling",
        "displayName-count-one";
        "Österreichischer Schilling",
        "displayName-count-other";
        "Österreichische Schilling",
        "symbol";
        "öS";
    }
    "AUD";
    {
        "displayName";
        "Australischer Dollar",
        "displayName-count-one";
        "Australischer Dollar",
        "displayName-count-other";
        "Australische Dollar",
        "symbol";
        "AU$",
        "symbol-alt-narrow";
        "$";
    }
    "AWG";
    {
        "displayName";
        "Aruba-Florin",
        "displayName-count-one";
        "Aruba-Florin",
        "displayName-count-other";
        "Aruba-Florin",
        "symbol";
        "AWG";
    }
    "AZM";
    {
        "displayName";
        "Aserbaidtschan-Manat (1993-2006)",
        "displayName-count-one";
        "Aserbaidtschan-Manat (1993-2006)",

```

```

        "displayName-count-other";
        "Aserbaidshan-Manat (1993-2006)",
        "symbol";
        "AZM";
    }
    "AZN";
    {
        "displayName";
        "Aserbaidshan-Manat",
        "displayName-count-one";
        "Aserbaidshan-Manat",
        "displayName-count-other";
        "Aserbaidshan-Manat",
        "symbol";
        "AZN";
    }
    "BAD";
    {
        "displayName";
        "Bosnien und Herzegowina Dinar (1992-1994)",
        "displayName-count-one";
        "Bosnien und Herzegowina Dinar (1992-1994)",
        "displayName-count-other";
        "Bosnien und Herzegowina Dinar (1992-1994)",
        "symbol";
        "BAD";
    }
    "BAM";
    {
        "displayName";
        "Bosnien und Herzegowina Konvertierbare Mark",
        "displayName-count-one";
        "Bosnien und Herzegowina Konvertierbare Mark",
        "displayName-count-other";
        "Bosnien und Herzegowina Konvertierbare Mark",
        "symbol";
        "BAM",
        "symbol-alt-narrow";
        "KM";
    }
    "BAN";
    {
        "displayName";
        "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
        "displayName-count-one";
        "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
        "displayName-count-other";
        "Bosnien und Herzegowina Neue Dinar (1994-1997)",
        "symbol";
        "BAN";
    }
    "BBD";
    {
        "displayName";
        "Barbados-Dollar",
        "displayName-count-one";
        "Barbados-Dollar",

```

```

        "displayName-count-other";
        "Barbados-Dollar",
        "symbol";
        "BBD",
        "symbol-alt-narrow";
        "$";
    }
    "BDT";
    {
        "displayName";
        "Bangladesch-Taka",
        "displayName-count-one";
        "Bangladesch-Taka",
        "displayName-count-other";
        "Bangladesch-Taka",
        "symbol";
        "BDT",
        "symbol-alt-narrow";
        "৳";
    }
    "BEC";
    {
        "displayName";
        "Belgischer Franc (konvertibel)",
        "displayName-count-one";
        "Belgischer Franc (konvertibel)",
        "displayName-count-other";
        "Belgische Franc (konvertibel)",
        "symbol";
        "BEC";
    }
    "BEF";
    {
        "displayName";
        "Belgischer Franc",
        "displayName-count-one";
        "Belgischer Franc",
        "displayName-count-other";
        "Belgische Franc",
        "symbol";
        "BEF";
    }
    "BEL";
    {
        "displayName";
        "Belgischer Finanz-Franc",
        "displayName-count-one";
        "Belgischer Finanz-Franc",
        "displayName-count-other";
        "Belgische Finanz-Franc",
        "symbol";
        "BEL";
    }
    "BGL";
    {
        "displayName";

```



```

        "Bulgarische Lew (1962-1999)",
        "displayName-count-one";
        "Bulgarische Lew (1962-1999)",
        "displayName-count-other";
        "Bulgarische Lew (1962-1999)",
        "symbol";
        "BGL";
    }
    "BGM";
    {
        "displayName";
        "Bulgarischer Lew (1952-1962)",
        "displayName-count-one";
        "Bulgarischer Lew (1952-1962)",
        "displayName-count-other";
        "Bulgarische Lew (1952-1962)",
        "symbol";
        "BGK";
    }
    "BGN";
    {
        "displayName";
        "Bulgarischer Lew",
        "displayName-count-one";
        "Bulgarischer Lew",
        "displayName-count-other";
        "Bulgarische Lew",
        "symbol";
        "BGN";
    }
    "BGO";
    {
        "displayName";
        "Bulgarischer Lew (1879-1952)",
        "displayName-count-one";
        "Bulgarischer Lew (1879-1952)",
        "displayName-count-other";
        "Bulgarische Lew (1879-1952)",
        "symbol";
        "BGJ";
    }
    "BHD";
    {
        "displayName";
        "Bahrain-Dinar",
        "displayName-count-one";
        "Bahrain-Dinar",
        "displayName-count-other";
        "Bahrain-Dinar",
        "symbol";
        "BHD";
    }
    "BIF";
    {
        "displayName";
        "Burundi-Franc",
        "displayName-count-one";

```

```

        "Burundi-Franc",
        "displayName-count-other";
        "Burundi-Francs",
        "symbol";
        "BIF";
    }
    "BMD";
    {
        "displayName";
        "Bermuda-Dollar",
        "displayName-count-one";
        "Bermuda-Dollar",
        "displayName-count-other";
        "Bermuda-Dollar",
        "symbol";
        "BMD",
        "symbol-alt-narrow";
        "$";
    }
    "BND";
    {
        "displayName";
        "Brunei-Dollar",
        "displayName-count-one";
        "Brunei-Dollar",
        "displayName-count-other";
        "Brunei-Dollar",
        "symbol";
        "BND",
        "symbol-alt-narrow";
        "$";
    }
    "BOB";
    {
        "displayName";
        "Bolivanischer Boliviano",
        "displayName-count-one";
        "Bolivanischer Boliviano",
        "displayName-count-other";
        "Bolivianische Bolivianos",
        "symbol";
        "BOB",
        "symbol-alt-narrow";
        "Bs";
    }
    "BOL";
    {
        "displayName";
        "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-one";
        "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-other";
        "Bolivianische Bolivianos (1863-1963)",
        "symbol";
        "BOL";
    }
    "BOP";

```

```

    {
      "displayName";
      "Bolivianischer Peso",
      "displayName-count-one";
      "Bolivianischer Peso",
      "displayName-count-other";
      "Bolivianische Peso",
      "symbol";
      "BOP";
    }
    "BOV";
    {
      "displayName";
      "Boliviansiche Mvdol",
      "displayName-count-one";
      "Boliviansiche Mvdol",
      "displayName-count-other";
      "Bolivianische Mvdol",
      "symbol";
      "BOV";
    }
    "BRB";
    {
      "displayName";
      "Brazilianischer Cruzeiro Novo (1967-1986)",
      "displayName-count-one";
      "Brazilianischer Cruzeiro Novo (1967-1986)",
      "displayName-count-other";
      "Brazilianische Cruzeiro Novo (1967-1986)",
      "symbol";
      "BRB";
    }
    "BRC";
    {
      "displayName";
      "Brazilianischer Cruzado (1986-1989)",
      "displayName-count-one";
      "Brazilianischer Cruzado (1986-1989)",
      "displayName-count-other";
      "Brazilianische Cruzado (1986-1989)",
      "symbol";
      "BRC";
    }
    "BRE";
    {
      "displayName";
      "Brazilianischer Cruzeiro (1990-1993)",
      "displayName-count-one";
      "Brazilianischer Cruzeiro (1990-1993)",
      "displayName-count-other";
      "Brazilianische Cruzeiro (1990-1993)",
      "symbol";
      "BRE";
    }
    "BRL";
    {
      "displayName";

```

```

        "Brasilianischer Real",
        "displayName-count-one";
        "Brasilianischer Real",
        "displayName-count-other";
        "Brasilianische Real",
        "symbol";
        "R$",
        "symbol-alt-narrow";
        "R$";
    }
    "BRN";
    {
        "displayName";
        "Brasilianischer Cruzado Novo (1989-1990)",
        "displayName-count-one";
        "Brasilianischer Cruzado Novo (1989-1990)",
        "displayName-count-other";
        "Brasilianische Cruzado Novo (1989-1990)",
        "symbol";
        "BRN";
    }
    "BRR";
    {
        "displayName";
        "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-one";
        "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-other";
        "Brasilianische Cruzeiro (1993-1994)",
        "symbol";
        "BRR";
    }
    "BRZ";
    {
        "displayName";
        "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-one";
        "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-other";
        "Brasilianischer Cruzeiro (1942-1967)",
        "symbol";
        "BRZ";
    }
    "BSD";
    {
        "displayName";
        "Bahamas-Dollar",
        "displayName-count-one";
        "Bahamas-Dollar",
        "displayName-count-other";
        "Bahamas-Dollar",
        "symbol";
        "BSD",
        "symbol-alt-narrow";
        "$";
    }
    "BTN";

```

```

    {
      "displayName";
      "Bhutan-Ngultrum",
      "displayName-count-one";
      "Bhutan-Ngultrum",
      "displayName-count-other";
      "Bhutan-Ngultrum",
      "symbol";
      "BTN";
    }
    "BUK";
    {
      "displayName";
      "Birmanischer Kyat",
      "displayName-count-one";
      "Birmanischer Kyat",
      "displayName-count-other";
      "Birmanische Kyat",
      "symbol";
      "BUK";
    }
    "BWP";
    {
      "displayName";
      "Botswanischer Pula",
      "displayName-count-one";
      "Botswanischer Pula",
      "displayName-count-other";
      "Botswanische Pula",
      "symbol";
      "BWP",
      "symbol-alt-narrow";
      "P";
    }
    "BYB";
    {
      "displayName";
      "Belarus-Rubel (1994-1999)",
      "displayName-count-one";
      "Belarus-Rubel (1994-1999)",
      "displayName-count-other";
      "Belarus-Rubel (1994-1999)",
      "symbol";
      "BYB";
    }
    "BYN";
    {
      "displayName";
      "Weißrussischer Rubel",
      "displayName-count-one";
      "Weißrussischer Rubel",
      "displayName-count-other";
      "Weißrussische Rubel",
      "symbol";
      "BYN",
      "symbol-alt-narrow";
      "p.";
    }
  
```

```

    }
    "BYR";
    {
        "displayName";
        "Weißrussischer Rubel (2000-2016)",
        "displayName-count-one";
        "Weißrussischer Rubel (2000-2016)",
        "displayName-count-other";
        "Weißrussische Rubel (2000-2016)",
        "symbol";
        "BYR";
    }
    "BZD";
    {
        "displayName";
        "Belize-Dollar",
        "displayName-count-one";
        "Belize-Dollar",
        "displayName-count-other";
        "Belize-Dollar",
        "symbol";
        "BZD",
        "symbol-alt-narrow";
        "$";
    }
    "CAD";
    {
        "displayName";
        "Kanadischer Dollar",
        "displayName-count-one";
        "Kanadischer Dollar",
        "displayName-count-other";
        "Kanadische Dollar",
        "symbol";
        "CA$",
        "symbol-alt-narrow";
        "$";
    }
    "CDF";
    {
        "displayName";
        "Kongo-Franc",
        "displayName-count-one";
        "Kongo-Franc",
        "displayName-count-other";
        "Kongo-Francs",
        "symbol";
        "CDF";
    }
    "CHE";
    {
        "displayName";
        "WIR-Euro",
        "displayName-count-one";
        "WIR-Euro",
        "displayName-count-other";
        "WIR-Euro",
    }

```

```
        "symbol";
        "CHE";
    }
    "CHF";
    {
        "displayName";
        "Schweizer Franken",
        "displayName-count-one";
        "Schweizer Franken",
        "displayName-count-other";
        "Schweizer Franken",
        "symbol";
        "CHF";
    }
    "CHW";
    {
        "displayName";
        "WIR Franken",
        "displayName-count-one";
        "WIR Franken",
        "displayName-count-other";
        "WIR Franken",
        "symbol";
        "CHW";
    }
    "CLE";
    {
        "displayName";
        "Chilenischer Escudo",
        "displayName-count-one";
        "Chilenischer Escudo",
        "displayName-count-other";
        "Chilenische Escudo",
        "symbol";
        "CLE";
    }
    "CLF";
    {
        "displayName";
        "Chilenische Unidades de Fomento",
        "displayName-count-one";
        "Chilenische Unidades de Fomento",
        "displayName-count-other";
        "Chilenische Unidades de Fomento",
        "symbol";
        "CLF";
    }
    "CLP";
    {
        "displayName";
        "Chilenischer Peso",
        "displayName-count-one";
        "Chilenischer Peso",
        "displayName-count-other";
        "Chilenische Pesos",
        "symbol";
        "CLP";
```

```

        "symbol-alt-narrow";
        "$";
    }
    "CNX";
    {
        "displayName";
        "Dollar der Chinesischen Volksbank",
        "displayName-count-one";
        "Dollar der Chinesischen Volksbank",
        "displayName-count-other";
        "Dollar der Chinesischen Volksbank",
        "symbol";
        "CNX";
    }
    "CNY";
    {
        "displayName";
        "Renminbi Yuan",
        "displayName-count-one";
        "Chinesischer Yuan",
        "displayName-count-other";
        "Renminbi Yuan",
        "symbol";
        "CN¥",
        "symbol-alt-narrow";
        "¥";
    }
    "COP";
    {
        "displayName";
        "Kolumbianischer Peso",
        "displayName-count-one";
        "Kolumbianischer Peso",
        "displayName-count-other";
        "Kolumbianische Pesos",
        "symbol";
        "COP",
        "symbol-alt-narrow";
        "$";
    }
    "COU";
    {
        "displayName";
        "Kolumbianische Unidades de valor real",
        "displayName-count-one";
        "Kolumbianische Unidad de valor real",
        "displayName-count-other";
        "Kolumbianische Unidades de valor real",
        "symbol";
        "COU";
    }
    "CRC";
    {
        "displayName";
        "Costa-Rica-Colón",
        "displayName-count-one";
        "Costa-Rica-Colón",

```



```

        "displayName-count-other";
        "Costa-Rica-Colón",
        "symbol";
        "CRC",
        "symbol-alt-narrow";
        "₡";
    }
    "CSD";
    {
        "displayName";
        "Serbischer Dinar (2002-2006)",
        "displayName-count-one";
        "Serbischer Dinar (2002-2006)",
        "displayName-count-other";
        "Serbische Dinar (2002-2006)",
        "symbol";
        "CSD";
    }
    "CSK";
    {
        "displayName";
        "Tschechoslowakische Krone",
        "displayName-count-one";
        "Tschechoslowakische Kronen",
        "displayName-count-other";
        "Tschechoslowakische Kronen",
        "symbol";
        "CSK";
    }
    "CUC";
    {
        "displayName";
        "Kubanischer Peso (konvertibel)",
        "displayName-count-one";
        "Kubanischer Peso (konvertibel)",
        "displayName-count-other";
        "Kubanische Pesos (konvertibel)",
        "symbol";
        "CUC",
        "symbol-alt-narrow";
        "Cub$";
    }
    "CUP";
    {
        "displayName";
        "Kubanischer Peso",
        "displayName-count-one";
        "Kubanischer Peso",
        "displayName-count-other";
        "Kubanische Pesos",
        "symbol";
        "CUP",
        "symbol-alt-narrow";
        "$";
    }
    "CVE";
    {

```

```

        "displayName";
        "Cabo-Verde-Escudo",
        "displayName-count-one";
        "Cabo-Verde-Escudo",
        "displayName-count-other";
        "Cabo-Verde-Escudos",
        "symbol";
        "CVE";
    }
    "CYP";
    {
        "displayName";
        "Zypern-Pfund",
        "displayName-count-one";
        "Zypern Pfund",
        "displayName-count-other";
        "Zypern Pfund",
        "symbol";
        "CYP";
    }
    "CZK";
    {
        "displayName";
        "Tschechische Krone",
        "displayName-count-one";
        "Tschechische Krone",
        "displayName-count-other";
        "Tschechische Kronen",
        "symbol";
        "CZK",
        "symbol-alt-narrow";
        "Kč";
    }
    "DDM";
    {
        "displayName";
        "Mark der DDR",
        "displayName-count-one";
        "Mark der DDR",
        "displayName-count-other";
        "Mark der DDR",
        "symbol";
        "DDM";
    }
    "DEM";
    {
        "displayName";
        "Deutsche Mark",
        "displayName-count-one";
        "Deutsche Mark",
        "displayName-count-other";
        "Deutsche Mark",
        "symbol";
        "DM";
    }
    "DJF";
    {

```

```

        "displayName";
        "Dschibuti-Franc",
        "displayName-count-one";
        "Dschibuti-Franc",
        "displayName-count-other";
        "Dschibuti-Franc",
        "symbol";
        "DJF";
    }
    "DKK";
    {
        "displayName";
        "Dänische Krone",
        "displayName-count-one";
        "Dänische Krone",
        "displayName-count-other";
        "Dänische Kronen",
        "symbol";
        "DKK",
        "symbol-alt-narrow";
        "kr";
    }
    "DOP";
    {
        "displayName";
        "Dominikanischer Peso",
        "displayName-count-one";
        "Dominikanischer Peso",
        "displayName-count-other";
        "Dominikanische Pesos",
        "symbol";
        "DOP",
        "symbol-alt-narrow";
        "$";
    }
    "DZD";
    {
        "displayName";
        "Algerischer Dinar",
        "displayName-count-one";
        "Algerischer Dinar",
        "displayName-count-other";
        "Algerische Dinar",
        "symbol";
        "DZD";
    }
    "ECS";
    {
        "displayName";
        "Ecuadorianischer Sucre",
        "displayName-count-one";
        "Ecuadorianischer Sucre",
        "displayName-count-other";
        "Ecuadorianische Sucre",
        "symbol";
        "ECS";
    }
}

```

```

"ECV";
{
  "displayName";
  "Verrechnungseinheit für Ecuador",
    "displayName-count-one";
  "Verrechnungseinheiten für Ecuador",
    "displayName-count-other";
  "Verrechnungseinheiten für Ecuador",
    "symbol";
  "ECV";
}
"EEK";
{
  "displayName";
  "Estnische Krone",
    "displayName-count-one";
  "Estnische Krone",
    "displayName-count-other";
  "Estnische Kronen",
    "symbol";
  "EEK";
}
"EGP";
{
  "displayName";
  "Ägyptisches Pfund",
    "displayName-count-one";
  "Ägyptisches Pfund",
    "displayName-count-other";
  "Ägyptische Pfund",
    "symbol";
  "EGP",
    "symbol-alt-narrow";
  "£";
}
"ERN";
{
  "displayName";
  "Eritreischer Nakfa",
    "displayName-count-one";
  "Eritreischer Nakfa",
    "displayName-count-other";
  "Eritreische Nakfa",
    "symbol";
  "ERN";
}
"ESA";
{
  "displayName";
  "Spanische Peseta (A-Konten)",
    "displayName-count-one";
  "Spanische Peseta (A-Konten)",
    "displayName-count-other";
  "Spanische Peseten (A-Konten)",
    "symbol";
  "ESA";
}

```

```
"ESB";
{
  "displayName";
  "Spanische Peseta (konvertibel)",
    "displayName-count-one";
  "Spanische Peseta (konvertibel)",
    "displayName-count-other";
  "Spanische Peseten (konvertibel)",
    "symbol";
  "ESB";
}
"ESP";
{
  "displayName";
  "Spanische Peseta",
    "displayName-count-one";
  "Spanische Peseta",
    "displayName-count-other";
  "Spanische Peseten",
    "symbol";
  "ESP",
    "symbol-alt-narrow";
  "₧";
}
"ETB";
{
  "displayName";
  "Äthiopischer Birr",
    "displayName-count-one";
  "Äthiopischer Birr",
    "displayName-count-other";
  "Äthiopische Birr",
    "symbol";
  "ETB";
}
"EUR";
{
  "displayName";
  "Euro",
    "displayName-count-one";
  "Euro",
    "displayName-count-other";
  "Euro",
    "symbol";
  "€",
    "symbol-alt-narrow";
  "€";
}
"FIM";
{
  "displayName";
  "Finnische Mark",
    "displayName-count-one";
  "Finnische Mark",
    "displayName-count-other";
  "Finnische Mark",
    "symbol";
}
```

```

        "FIM";
    }
    "FJD";
    {
        "displayName";
        "Fidschi-Dollar",
        "displayName-count-one";
        "Fidschi-Dollar",
        "displayName-count-other";
        "Fidschi-Dollar",
        "symbol";
        "FJD",
        "symbol-alt-narrow";
        "$";
    }
    "FKP";
    {
        "displayName";
        "Falkland-Pfund",
        "displayName-count-one";
        "Falkland-Pfund",
        "displayName-count-other";
        "Falkland-Pfund",
        "symbol";
        "FKP",
        "symbol-alt-narrow";
        "Fl£";
    }
    "FRF";
    {
        "displayName";
        "Französischer Franc",
        "displayName-count-one";
        "Französischer Franc",
        "displayName-count-other";
        "Französische Franc",
        "symbol";
        "FRF";
    }
    "GBP";
    {
        "displayName";
        "Britisches Pfund",
        "displayName-count-one";
        "Britisches Pfund",
        "displayName-count-other";
        "Britische Pfund",
        "symbol";
        "£",
        "symbol-alt-narrow";
        "£";
    }
    "GEK";
    {
        "displayName";
        "Georgischer Kupon Larit",
        "displayName-count-one";

```

```

        "Georgischer Kupon Larit",
        "displayName-count-other";
        "Georgische Kupon Larit",
        "symbol";
        "GEK";
    }
    "GEL";
    {
        "displayName";
        "Georgischer Lari",
        "displayName-count-one";
        "Georgischer Lari",
        "displayName-count-other";
        "Georgische Lari",
        "symbol";
        "GEL",
        "symbol-alt-narrow";
        "ლ",
        "symbol-alt-variant";
        "ლ";
    }
    "GHC";
    {
        "displayName";
        "Ghanaischer Cedi (1979-2007)",
        "displayName-count-one";
        "Ghanaischer Cedi (1979-2007)",
        "displayName-count-other";
        "Ghanaische Cedi (1979-2007)",
        "symbol";
        "GHC";
    }
    "GHS";
    {
        "displayName";
        "Ghanaischer Cedi",
        "displayName-count-one";
        "Ghanaischer Cedi",
        "displayName-count-other";
        "Ghanaische Cedi",
        "symbol";
        "GHS";
    }
    "GIP";
    {
        "displayName";
        "Gibraltar-Pfund",
        "displayName-count-one";
        "Gibraltar-Pfund",
        "displayName-count-other";
        "Gibraltar Pfund",
        "symbol";
        "GIP",
        "symbol-alt-narrow";
        "£";
    }
    "GMD";

```

```

    {
      "displayName";
      "Gambia-Dalasi",
      "displayName-count-one";
      "Gambia-Dalasi",
      "displayName-count-other";
      "Gambia-Dalasi",
      "symbol";
      "GMD";
    }
    "GNF";
    {
      "displayName";
      "Guinea-Franc",
      "displayName-count-one";
      "Guinea-Franc",
      "displayName-count-other";
      "Guinea-Franc",
      "symbol";
      "GNF",
      "symbol-alt-narrow";
      "F.G.";
    }
    "GNS";
    {
      "displayName";
      "Guineischer Syli",
      "displayName-count-one";
      "Guineischer Syli",
      "displayName-count-other";
      "Guineische Syli",
      "symbol";
      "GNS";
    }
    "GQE";
    {
      "displayName";
      "Äquatorialguinea-Ekwele",
      "displayName-count-one";
      "Äquatorialguinea-Ekwele",
      "displayName-count-other";
      "Äquatorialguinea-Ekwele",
      "symbol";
      "GQE";
    }
    "GRD";
    {
      "displayName";
      "Griechische Drachme",
      "displayName-count-one";
      "Griechische Drachme",
      "displayName-count-other";
      "Griechische Drachmen",
      "symbol";
      "GRD";
    }
    "GTQ";

```



```

    {
      "displayName";
      "Guatemaltekinscher Quetzal",
        "displayName-count-one";
      "Guatemaltekinscher Quetzal",
        "displayName-count-other";
      "Guatemaltekinsche Quetzales",
        "symbol";
      "GTQ",
        "symbol-alt-narrow";
      "Q";
    }
    "GWE";
    {
      "displayName";
      "Portugiesisch Guinea Escudo",
        "displayName-count-one";
      "Portugiesisch Guinea Escudo",
        "displayName-count-other";
      "Portugiesisch Guinea Escudo",
        "symbol";
      "GWE";
    }
    "GWP";
    {
      "displayName";
      "Guinea-Bissau Peso",
        "displayName-count-one";
      "Guinea-Bissau Peso",
        "displayName-count-other";
      "Guinea-Bissau Pesos",
        "symbol";
      "GWP";
    }
    "GYD";
    {
      "displayName";
      "Guyana-Dollar",
        "displayName-count-one";
      "Guyana-Dollar",
        "displayName-count-other";
      "Guyana-Dollar",
        "symbol";
      "GYD",
        "symbol-alt-narrow";
      "$";
    }
    "HKD";
    {
      "displayName";
      "Hongkong-Dollar",
        "displayName-count-one";
      "Hongkong-Dollar",
        "displayName-count-other";
      "Hongkong-Dollar",
        "symbol";
      "HK$";
    }
  }

```

```

        "symbol-alt-narrow";
        "$";
    }
    "HNL";
    {
        "displayName";
        "Honduras-Lempira",
        "displayName-count-one";
        "Honduras-Lempira",
        "displayName-count-other";
        "Honduras-Lempira",
        "symbol";
        "HNL",
        "symbol-alt-narrow";
        "L";
    }
    "HRD";
    {
        "displayName";
        "Kroatischer Dinar",
        "displayName-count-one";
        "Kroatischer Dinar",
        "displayName-count-other";
        "Kroatische Dinar",
        "symbol";
        "HRD";
    }
    "HRK";
    {
        "displayName";
        "Kroatischer Kuna",
        "displayName-count-one";
        "Kroatischer Kuna",
        "displayName-count-other";
        "Kroatische Kuna",
        "symbol";
        "HRK",
        "symbol-alt-narrow";
        "kn";
    }
    "HTG";
    {
        "displayName";
        "Haitianische Gourde",
        "displayName-count-one";
        "Haitianische Gourde",
        "displayName-count-other";
        "Haitianische Gourdes",
        "symbol";
        "HTG";
    }
    "HUF";
    {
        "displayName";
        "Ungarischer Forint",
        "displayName-count-one";
        "Ungarischer Forint",

```

```

        "displayName-count-other";
        "Ungarische Forint",
        "symbol";
        "HUF",
        "symbol-alt-narrow";
        "Ft";
    }
    "IDR";
    {
        "displayName";
        "Indonesische Rupiah",
        "displayName-count-one";
        "Indonesische Rupiah",
        "displayName-count-other";
        "Indonesische Rupiah",
        "symbol";
        "IDR",
        "symbol-alt-narrow";
        "Rp";
    }
    "IEP";
    {
        "displayName";
        "Irisches Pfund",
        "displayName-count-one";
        "Irisches Pfund",
        "displayName-count-other";
        "Irische Pfund",
        "symbol";
        "IEP";
    }
    "ILP";
    {
        "displayName";
        "Israelisches Pfund",
        "displayName-count-one";
        "Israelisches Pfund",
        "displayName-count-other";
        "Israelische Pfund",
        "symbol";
        "ILP";
    }
    "ILR";
    {
        "displayName";
        "Israelischer Schekel (1980-1985)",
        "displayName-count-one";
        "Israelischer Schekel (1980-1985)",
        "displayName-count-other";
        "Israelische Schekel (1980-1985)";
    }
    "ILS";
    {
        "displayName";
        "Israelischer Neuer Schekel",
        "displayName-count-one";
        "Israelischer Neuer Schekel",

```

```

        "displayName-count-other";
        "Israelische Neue Schekel",
        "symbol";
        "₪",
        "symbol-alt-narrow";
        "₪";
    }
    "INR";
    {
        "displayName";
        "Indische Rupie",
        "displayName-count-one";
        "Indische Rupie",
        "displayName-count-other";
        "Indische Rupien",
        "symbol";
        "₹",
        "symbol-alt-narrow";
        "₹";
    }
    "IQD";
    {
        "displayName";
        "Irakischer Dinar",
        "displayName-count-one";
        "Irakischer Dinar",
        "displayName-count-other";
        "Irakische Dinar",
        "symbol";
        "IQD";
    }
    "IRR";
    {
        "displayName";
        "Iranischer Rial",
        "displayName-count-one";
        "Iranischer Rial",
        "displayName-count-other";
        "Iranische Rial",
        "symbol";
        "IRR";
    }
    "ISJ";
    {
        "displayName";
        "Isländische Krone (1918-1981)",
        "displayName-count-one";
        "Isländische Krone (1918-1981)",
        "displayName-count-other";
        "Isländische Kronen (1918-1981)";
    }
    "ISK";
    {
        "displayName";
        "Isländische Krone",
        "displayName-count-one";
        "Isländische Krone",

```

```

        "displayName-count-other";
        "Isländische Kronen",
        "symbol";
        "ISK",
        "symbol-alt-narrow";
        "kr";
    }
    "ITL";
    {
        "displayName";
        "Italienische Lira",
        "displayName-count-one";
        "Italienische Lira",
        "displayName-count-other";
        "Italienische Lire",
        "symbol";
        "ITL";
    }
    "JMD";
    {
        "displayName";
        "Jamaika-Dollar",
        "displayName-count-one";
        "Jamaika-Dollar",
        "displayName-count-other";
        "Jamaika-Dollar",
        "symbol";
        "JMD",
        "symbol-alt-narrow";
        "$";
    }
    "JOD";
    {
        "displayName";
        "Jordanischer Dinar",
        "displayName-count-one";
        "Jordanischer Dinar",
        "displayName-count-other";
        "Jordanische Dinar",
        "symbol";
        "JOD";
    }
    "JPY";
    {
        "displayName";
        "Japanischer Yen",
        "displayName-count-one";
        "Japanischer Yen",
        "displayName-count-other";
        "Japanische Yen",
        "symbol";
        "¥",
        "symbol-alt-narrow";
        "¥";
    }
    "KES";
    {

```

```

        "displayName";
        "Kenia-Schilling",
        "displayName-count-one";
        "Kenia-Schilling",
        "displayName-count-other";
        "Kenia-Schilling",
        "symbol";
        "KES";
    }
    "KGS";
    {
        "displayName";
        "Kirgisischer Som",
        "displayName-count-one";
        "Kirgisischer Som",
        "displayName-count-other";
        "Kirgisische Som",
        "symbol";
        "KGS";
    }
    "KHR";
    {
        "displayName";
        "Kambodschanischer Riel",
        "displayName-count-one";
        "Kambodschanischer Riel",
        "displayName-count-other";
        "Kambodschanische Riel",
        "symbol";
        "KHR",
        "symbol-alt-narrow";
        "៛";
    }
    "KMF";
    {
        "displayName";
        "Komoren-Franc",
        "displayName-count-one";
        "Komoren-Franc",
        "displayName-count-other";
        "Komoren-Francs",
        "symbol";
        "KMF",
        "symbol-alt-narrow";
        "FC";
    }
    "KPW";
    {
        "displayName";
        "Nordkoreanischer Won",
        "displayName-count-one";
        "Nordkoreanischer Won",
        "displayName-count-other";
        "Nordkoreanische Won",
        "symbol";
        "KPW",

```

```

        "symbol-alt-narrow";
        "₩";
    }
    "KRH";
    {
        "displayName";
        "Südkoreanischer Hwan (1953-1962)",
        "displayName-count-one";
        "Südkoreanischer Hwan (1953-1962)",
        "displayName-count-other";
        "Südkoreanischer Hwan (1953-1962)",
        "symbol";
        "KRH";
    }
    "KRO";
    {
        "displayName";
        "Südkoreanischer Won (1945-1953)",
        "displayName-count-one";
        "Südkoreanischer Won (1945-1953)",
        "displayName-count-other";
        "Südkoreanischer Won (1945-1953)",
        "symbol";
        "KRO";
    }
    "KRW";
    {
        "displayName";
        "Südkoreanischer Won",
        "displayName-count-one";
        "Südkoreanischer Won",
        "displayName-count-other";
        "Südkoreanische Won",
        "symbol";
        "₩",
        "symbol-alt-narrow";
        "₩";
    }
    "KWD";
    {
        "displayName";
        "Kuwait-Dinar",
        "displayName-count-one";
        "Kuwait-Dinar",
        "displayName-count-other";
        "Kuwait-Dinar",
        "symbol";
        "KWD";
    }
    "KYD";
    {
        "displayName";
        "Kaiman-Dollar",
        "displayName-count-one";
        "Kaiman-Dollar",
        "displayName-count-other";
        "Kaiman-Dollar",

```

```

        "symbol";
        "KYD",
        "symbol-alt-narrow";
        "$";
    }
    "KZT";
    {
        "displayName";
        "Kasachischer Tenge",
        "displayName-count-one";
        "Kasachischer Tenge",
        "displayName-count-other";
        "Kasachische Tenge",
        "symbol";
        "KZT",
        "symbol-alt-narrow";
        "Т";
    }
    "LAK";
    {
        "displayName";
        "Laotischer Kip",
        "displayName-count-one";
        "Laotischer Kip",
        "displayName-count-other";
        "Laotische Kip",
        "symbol";
        "LAK",
        "symbol-alt-narrow";
        "₭";
    }
    "LBP";
    {
        "displayName";
        "Libanesisches Pfund",
        "displayName-count-one";
        "Libanesisches Pfund",
        "displayName-count-other";
        "Libanesische Pfund",
        "symbol";
        "LBP",
        "symbol-alt-narrow";
        "L£";
    }
    "LKR";
    {
        "displayName";
        "Sri-Lanka-Rupie",
        "displayName-count-one";
        "Sri-Lanka-Rupie",
        "displayName-count-other";
        "Sri-Lanka-Rupien",
        "symbol";
        "LKR",
        "symbol-alt-narrow";
        "Rs";
    }
}

```



```
"LRD";
{
  "displayName";
  "Liberianischer Dollar",
    "displayName-count-one";
  "Liberianischer Dollar",
    "displayName-count-other";
  "Liberianische Dollar",
    "symbol";
  "LRD",
    "symbol-alt-narrow";
  "$";
}
"LSL";
{
  "displayName";
  "Loti",
    "displayName-count-one";
  "Loti",
    "displayName-count-other";
  "Loti",
    "symbol";
  "LSL";
}
"LTL";
{
  "displayName";
  "Litauischer Litas",
    "displayName-count-one";
  "Litauischer Litas",
    "displayName-count-other";
  "Litauische Litas",
    "symbol";
  "LTL",
    "symbol-alt-narrow";
  "Lt";
}
"LTT";
{
  "displayName";
  "Litauischer Talonas",
    "displayName-count-one";
  "Litauische Talonas",
    "displayName-count-other";
  "Litauische Talonas",
    "symbol";
  "LTT";
}
"LUC";
{
  "displayName";
  "Luxemburgischer Franc (konvertibel)",
    "displayName-count-one";
  "Luxemburgische Franc (konvertibel)",
    "displayName-count-other";
  "Luxemburgische Franc (konvertibel)",
    "symbol";
}
```

```

        "LUC";
    }
    "LUF";
    {
        "displayName";
        "Luxemburgischer Franc",
        "displayName-count-one";
        "Luxemburgische Franc",
        "displayName-count-other";
        "Luxemburgische Franc",
        "symbol";
        "LUF";
    }
    "LUL";
    {
        "displayName";
        "Luxemburgischer Finanz-Franc",
        "displayName-count-one";
        "Luxemburgische Finanz-Franc",
        "displayName-count-other";
        "Luxemburgische Finanz-Franc",
        "symbol";
        "LUL";
    }
    "LVL";
    {
        "displayName";
        "Lettischer Lats",
        "displayName-count-one";
        "Lettischer Lats",
        "displayName-count-other";
        "Lettische Lats",
        "symbol";
        "LVL",
        "symbol-alt-narrow";
        "Ls";
    }
    "LVR";
    {
        "displayName";
        "Lettischer Rubel",
        "displayName-count-one";
        "Lettische Rubel",
        "displayName-count-other";
        "Lettische Rubel",
        "symbol";
        "LVR";
    }
    "LYD";
    {
        "displayName";
        "Libyscher Dinar",
        "displayName-count-one";
        "Libyscher Dinar",
        "displayName-count-other";
        "Libysche Dinar",
        "symbol";
    }

```

```

        "LYD";
    }
    "MAD";
    {
        "displayName";
        "Marokkanischer Dirham",
        "displayName-count-one";
        "Marokkanischer Dirham",
        "displayName-count-other";
        "Marokkanische Dirham",
        "symbol";
        "MAD";
    }
    "MAF";
    {
        "displayName";
        "Marokkanischer Franc",
        "displayName-count-one";
        "Marokkanische Franc",
        "displayName-count-other";
        "Marokkanische Franc",
        "symbol";
        "MAF";
    }
    "MCF";
    {
        "displayName";
        "Monegassischer Franc",
        "displayName-count-one";
        "Monegassischer Franc",
        "displayName-count-other";
        "Monegassische Franc",
        "symbol";
        "MCF";
    }
    "MDC";
    {
        "displayName";
        "Moldau-Cupon",
        "displayName-count-one";
        "Moldau-Cupon",
        "displayName-count-other";
        "Moldau-Cupon",
        "symbol";
        "MDC";
    }
    "MDL";
    {
        "displayName";
        "Moldau-Leu",
        "displayName-count-one";
        "Moldau-Leu",
        "displayName-count-other";
        "Moldau-Leu",
        "symbol";
        "MDL";
    }
}

```

```

    "MGA";
    {
      "displayName";
      "Madagaskar-Ariary",
        "displayName-count-one";
      "Madagaskar-Ariary",
        "displayName-count-other";
      "Madagaskar-Ariary",
        "symbol";
      "MGA",
        "symbol-alt-narrow";
      "Ar";
    }
    "MGF";
    {
      "displayName";
      "Madagaskar-Franc",
        "displayName-count-one";
      "Madagaskar-Franc",
        "displayName-count-other";
      "Madagaskar-Franc",
        "symbol";
      "MGF";
    }
    "MKD";
    {
      "displayName";
      "Mazedonischer Denar",
        "displayName-count-one";
      "Mazedonischer Denar",
        "displayName-count-other";
      "Mazedonische Denari",
        "symbol";
      "MKD";
    }
    "MKN";
    {
      "displayName";
      "Mazedonischer Denar (1992-1993)",
        "displayName-count-one";
      "Mazedonischer Denar (1992-1993)",
        "displayName-count-other";
      "Mazedonische Denar (1992-1993)",
        "symbol";
      "MKN";
    }
    "MLF";
    {
      "displayName";
      "Malischer Franc",
        "displayName-count-one";
      "Malische Franc",
        "displayName-count-other";
      "Malische Franc",
        "symbol";
      "MLF";
    }
  }

```

```

"MMK";
{
  "displayName";
  "Myanmarischer Kyat",
    "displayName-count-one";
  "Myanmarischer Kyat",
    "displayName-count-other";
  "Myanmarische Kyat",
    "symbol";
  "MMK",
    "symbol-alt-narrow";
  "K";
}
"MNT";
{
  "displayName";
  "Mongolischer Tögrög",
    "displayName-count-one";
  "Mongolischer Tögrög",
    "displayName-count-other";
  "Mongolische Tögrög",
    "symbol";
  "MNT",
    "symbol-alt-narrow";
  "₮";
}
"MOP";
{
  "displayName";
  "Macao-Pataca",
    "displayName-count-one";
  "Macao-Pataca",
    "displayName-count-other";
  "Macao-Pataca",
    "symbol";
  "MOP";
}
"MRO";
{
  "displayName";
  "Mauretanischer Ouguiya",
    "displayName-count-one";
  "Mauretanischer Ouguiya",
    "displayName-count-other";
  "Mauretanische Ouguiya",
    "symbol";
  "MRO";
}
"MTL";
{
  "displayName";
  "Maltesische Lira",
    "displayName-count-one";
  "Maltesische Lira",
    "displayName-count-other";
  "Maltesische Lira",
    "symbol";
}

```

```

        "MTL";
    }
    "MTP";
    {
        "displayName";
        "Maltesisches Pfund",
        "displayName-count-one";
        "Maltesische Pfund",
        "displayName-count-other";
        "Maltesische Pfund",
        "symbol";
        "MTP";
    }
    "MUR";
    {
        "displayName";
        "Mauritius-Rupie",
        "displayName-count-one";
        "Mauritius-Rupie",
        "displayName-count-other";
        "Mauritius-Rupien",
        "symbol";
        "MUR",
        "symbol-alt-narrow";
        "Rs";
    }
    "MVP";
    {
        "displayName";
        "Malediven-Rupie (alt)",
        "displayName-count-one";
        "Malediven-Rupie (alt)",
        "displayName-count-other";
        "Malediven-Rupien (alt)";
    }
    "MVR";
    {
        "displayName";
        "Malediven-Rufiyaa",
        "displayName-count-one";
        "Malediven-Rufiyaa",
        "displayName-count-other";
        "Malediven-Rupien",
        "symbol";
        "MVR";
    }
    "MWK";
    {
        "displayName";
        "Malawi-Kwacha",
        "displayName-count-one";
        "Malawi-Kwacha",
        "displayName-count-other";
        "Malawi-Kwacha",
        "symbol";
        "MWK";
    }
}

```

```

"MXN";
{
  "displayName";
  "Mexikanischer Peso",
    "displayName-count-one";
  "Mexikanischer Peso",
    "displayName-count-other";
  "Mexikanische Pesos",
    "symbol";
  "MX$",
    "symbol-alt-narrow";
  "$";
}
"MXP";
{
  "displayName";
  "Mexikanischer Silber-Peso (1861-1992)",
    "displayName-count-one";
  "Mexikanische Silber-Peso (1861-1992)",
    "displayName-count-other";
  "Mexikanische Silber-Pesos (1861-1992)",
    "symbol";
  "MXP";
}
"MXV";
{
  "displayName";
  "Mexicanischer Unidad de Inversion (UDI)",
    "displayName-count-one";
  "Mexicanischer Unidad de Inversion (UDI)",
    "displayName-count-other";
  "Mexikanische Unidad de Inversion (UDI)",
    "symbol";
  "MXV";
}
"MYR";
{
  "displayName";
  "Malaysischer Ringgit",
    "displayName-count-one";
  "Malaysischer Ringgit",
    "displayName-count-other";
  "Malaysische Ringgit",
    "symbol";
  "MYR",
    "symbol-alt-narrow";
  "RM";
}
"MZE";
{
  "displayName";
  "Mosambikanischer Escudo",
    "displayName-count-one";
  "Mozambikanische Escudo",
    "displayName-count-other";
  "Mozambikanische Escudo",
    "symbol";
}

```

```

        "MZE";
    }
    "MZM";
    {
        "displayName";
        "Mosambikanischer Metical (1980-2006)",
        "displayName-count-one";
        "Mosambikanischer Metical (1980-2006)",
        "displayName-count-other";
        "Mosambikanische Meticaïs (1980-2006)",
        "symbol";
        "MZM";
    }
    "MZN";
    {
        "displayName";
        "Mosambikanischer Metical",
        "displayName-count-one";
        "Mosambikanischer Metical",
        "displayName-count-other";
        "Mosambikanische Meticaïs",
        "symbol";
        "MZN";
    }
    "NAD";
    {
        "displayName";
        "Namibia-Dollar",
        "displayName-count-one";
        "Namibia-Dollar",
        "displayName-count-other";
        "Namibia-Dollar",
        "symbol";
        "NAD",
        "symbol-alt-narrow";
        "$";
    }
    "NGN";
    {
        "displayName";
        "Nigerianischer Naira",
        "displayName-count-one";
        "Nigerianischer Naira",
        "displayName-count-other";
        "Nigerianische Naira",
        "symbol";
        "NGN",
        "symbol-alt-narrow";
        "₦";
    }
    "NIC";
    {
        "displayName";
        "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-one";
        "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-other";
    }

```



```

        "Nicaraguanische Córdoba (1988-1991)",
        "symbol";
        "NIC";
    }
    "NIO";
    {
        "displayName";
        "Nicaragua-Córdoba",
        "displayName-count-one";
        "Nicaragua-Córdoba",
        "displayName-count-other";
        "Nicaragua-Córdobas",
        "symbol";
        "NIO",
        "symbol-alt-narrow";
        "C$";
    }
    "NLG";
    {
        "displayName";
        "Niederländischer Gulden",
        "displayName-count-one";
        "Niederländischer Gulden",
        "displayName-count-other";
        "Niederländische Gulden",
        "symbol";
        "NLG";
    }
    "NOK";
    {
        "displayName";
        "Norwegische Krone",
        "displayName-count-one";
        "Norwegische Krone",
        "displayName-count-other";
        "Norwegische Kronen",
        "symbol";
        "NOK",
        "symbol-alt-narrow";
        "kr";
    }
    "NPR";
    {
        "displayName";
        "Nepalesische Rupie",
        "displayName-count-one";
        "Nepalesische Rupie",
        "displayName-count-other";
        "Nepalesische Rupien",
        "symbol";
        "NPR",
        "symbol-alt-narrow";
        "Rs";
    }
    "NZD";
    {
        "displayName";

```

```

        "Neuseeland-Dollar",
        "displayName-count-one";
        "Neuseeland-Dollar",
        "displayName-count-other";
        "Neuseeland-Dollar",
        "symbol";
        "NZ$",
        "symbol-alt-narrow";
        "$";
    }
    "OMR";
    {
        "displayName";
        "Omanischer Rial",
        "displayName-count-one";
        "Omanischer Rial",
        "displayName-count-other";
        "Omanische Rials",
        "symbol";
        "OMR";
    }
    "PAB";
    {
        "displayName";
        "Panamaischer Balboa",
        "displayName-count-one";
        "Panamaischer Balboa",
        "displayName-count-other";
        "Panamaische Balboas",
        "symbol";
        "PAB";
    }
    "PEI";
    {
        "displayName";
        "Peruanischer Inti",
        "displayName-count-one";
        "Peruanische Inti",
        "displayName-count-other";
        "Peruanische Inti",
        "symbol";
        "PEI";
    }
    "PEN";
    {
        "displayName";
        "Peruanischer Sol",
        "displayName-count-one";
        "Peruanischer Sol",
        "displayName-count-other";
        "Peruanische Sol",
        "symbol";
        "PEN";
    }
    "PES";
    {
        "displayName";

```

```

        "Peruanischer Sol (1863-1965)",
        "displayName-count-one";
        "Peruanischer Sol (1863-1965)",
        "displayName-count-other";
        "Peruanische Sol (1863-1965)",
        "symbol";
        "PES";
    }
    "PGK";
    {
        "displayName";
        "Papua-Neuguineischer Kina",
        "displayName-count-one";
        "Papua-Neuguineischer Kina",
        "displayName-count-other";
        "Papua-Neuguineische Kina",
        "symbol";
        "PGK";
    }
    "PHP";
    {
        "displayName";
        "Philippinischer Peso",
        "displayName-count-one";
        "Philippinischer Peso",
        "displayName-count-other";
        "Philippinische Pesos",
        "symbol";
        "PHP",
        "symbol-alt-narrow";
        "₱";
    }
    "PKR";
    {
        "displayName";
        "Pakistanische Rupie",
        "displayName-count-one";
        "Pakistanische Rupie",
        "displayName-count-other";
        "Pakistanische Rupien",
        "symbol";
        "PKR",
        "symbol-alt-narrow";
        "Rs";
    }
    "PLN";
    {
        "displayName";
        "Polnischer Złoty",
        "displayName-count-one";
        "Polnischer Złoty",
        "displayName-count-other";
        "Polnische Złoty",
        "symbol";
        "PLN",
        "symbol-alt-narrow";
        "zł";
    }

```

```

    }
    "PLZ";
    {
        "displayName";
        "Polnischer Zloty (1950-1995)",
        "displayName-count-one";
        "Polnischer Zloty (1950-1995)",
        "displayName-count-other";
        "Polnische Zloty (1950-1995)",
        "symbol";
        "PLZ";
    }
    "PTE";
    {
        "displayName";
        "Portugiesischer Escudo",
        "displayName-count-one";
        "Portugiesische Escudo",
        "displayName-count-other";
        "Portugiesische Escudo",
        "symbol";
        "PTE";
    }
    "PYG";
    {
        "displayName";
        "Paraguayischer Guaraní",
        "displayName-count-one";
        "Paraguayischer Guaraní",
        "displayName-count-other";
        "Paraguayische Guaraníes",
        "symbol";
        "PYG",
        "symbol-alt-narrow";
        "₲";
    }
    "QAR";
    {
        "displayName";
        "Katar-Riyal",
        "displayName-count-one";
        "Katar-Riyal",
        "displayName-count-other";
        "Katar-Riyal",
        "symbol";
        "QAR";
    }
    "RHD";
    {
        "displayName";
        "Rhodesischer Dollar",
        "displayName-count-one";
        "Rhodesische Dollar",
        "displayName-count-other";
        "Rhodesische Dollar",
        "symbol";
        "RHD";
    }

```

```

    }
    "ROL";
    {
        "displayName";
        "Rumänischer Leu (1952-2006)",
        "displayName-count-one";
        "Rumänischer Leu (1952-2006)",
        "displayName-count-other";
        "Rumänische Leu (1952-2006)",
        "symbol";
        "ROL";
    }
    "RON";
    {
        "displayName";
        "Rumänischer Leu",
        "displayName-count-one";
        "Rumänischer Leu",
        "displayName-count-other";
        "Rumänische Leu",
        "symbol";
        "RON",
        "symbol-alt-narrow";
        "L";
    }
    "RSD";
    {
        "displayName";
        "Serbischer Dinar",
        "displayName-count-one";
        "Serbischer Dinar",
        "displayName-count-other";
        "Serbische Dinaren",
        "symbol";
        "RSD";
    }
    "RUB";
    {
        "displayName";
        "Russischer Rubel",
        "displayName-count-one";
        "Russischer Rubel",
        "displayName-count-other";
        "Russische Rubel",
        "symbol";
        "RUB",
        "symbol-alt-narrow";
        "P";
    }
    "RUR";
    {
        "displayName";
        "Russischer Rubel (1991-1998)",
        "displayName-count-one";
        "Russischer Rubel (1991-1998)",
        "displayName-count-other";
        "Russische Rubel (1991-1998)",

```

```

        "symbol";
        "RUR",
        "symbol-alt-narrow";
        "p.";
    }
    "RWF";
    {
        "displayName";
        "Ruanda-Franc",
        "displayName-count-one";
        "Ruanda-Franc",
        "displayName-count-other";
        "Ruanda-Francis",
        "symbol";
        "RWF",
        "symbol-alt-narrow";
        "F.Rw";
    }
    "SAR";
    {
        "displayName";
        "Saudi-Rial",
        "displayName-count-one";
        "Saudi-Rial",
        "displayName-count-other";
        "Saudi-Rial",
        "symbol";
        "SAR";
    }
    "SBD";
    {
        "displayName";
        "Salomonen-Dollar",
        "displayName-count-one";
        "Salomonen-Dollar",
        "displayName-count-other";
        "Salomonen-Dollar",
        "symbol";
        "SBD",
        "symbol-alt-narrow";
        "$";
    }
    "SCR";
    {
        "displayName";
        "Seychellen-Rupie",
        "displayName-count-one";
        "Seychellen-Rupie",
        "displayName-count-other";
        "Seychellen-Rupien",
        "symbol";
        "SCR";
    }
    "SDD";
    {
        "displayName";
        "Sudanesischer Dinar (1992-2007)",

```

```

        "displayName-count-one";
        "Sudanesischer Dinar (1992-2007)",
        "displayName-count-other";
        "Sudanesische Dinar (1992-2007)",
        "symbol";
        "SDD";
    }
    "SDG";
    {
        "displayName";
        "Sudanesisches Pfund",
        "displayName-count-one";
        "Sudanesisches Pfund",
        "displayName-count-other";
        "Sudanesische Pfund",
        "symbol";
        "SDG";
    }
    "SDP";
    {
        "displayName";
        "Sudanesisches Pfund (1957-1998)",
        "displayName-count-one";
        "Sudanesisches Pfund (1957-1998)",
        "displayName-count-other";
        "Sudanesische Pfund (1957-1998)",
        "symbol";
        "SDP";
    }
    "SEK";
    {
        "displayName";
        "Schwedische Krone",
        "displayName-count-one";
        "Schwedische Krone",
        "displayName-count-other";
        "Schwedische Kronen",
        "symbol";
        "SEK",
        "symbol-alt-narrow";
        "kr";
    }
    "SGD";
    {
        "displayName";
        "Singapur-Dollar",
        "displayName-count-one";
        "Singapur-Dollar",
        "displayName-count-other";
        "Singapur-Dollar",
        "symbol";
        "SGD",
        "symbol-alt-narrow";
        "$";
    }
    "SHP";
    {

```

```

        "displayName";
        "St. Helena-Pfund",
        "displayName-count-one";
        "St. Helena-Pfund",
        "displayName-count-other";
        "St. Helena-Pfund",
        "symbol";
        "SHP",
        "symbol-alt-narrow";
        "£";
    }
    "SIT";
    {
        "displayName";
        "Slowenischer Tolar",
        "displayName-count-one";
        "Slowenischer Tolar",
        "displayName-count-other";
        "Slowenische Tolar",
        "symbol";
        "SIT";
    }
    "SKK";
    {
        "displayName";
        "Slowakische Krone",
        "displayName-count-one";
        "Slowakische Kronen",
        "displayName-count-other";
        "Slowakische Kronen",
        "symbol";
        "SKK";
    }
    "SLL";
    {
        "displayName";
        "Sierra-leonischer Leone",
        "displayName-count-one";
        "Sierra-leonischer Leone",
        "displayName-count-other";
        "Sierra-leonische Leones",
        "symbol";
        "SLL";
    }
    "SOS";
    {
        "displayName";
        "Somalia-Schilling",
        "displayName-count-one";
        "Somalia-Schilling",
        "displayName-count-other";
        "Somalia-Schilling",
        "symbol";
        "SOS";
    }
    "SRD";
    {

```



```

        "displayName";
        "Suriname-Dollar",
            "displayName-count-one";
        "Suriname-Dollar",
            "displayName-count-other";
        "Suriname-Dollar",
            "symbol";
        "SRD",
            "symbol-alt-narrow";
        "$";
    }
    "SRG";
    {
        "displayName";
        "Suriname Gulden",
            "displayName-count-one";
        "Suriname-Gulden",
            "displayName-count-other";
        "Suriname-Gulden",
            "symbol";
        "SRG";
    }
    "SSP";
    {
        "displayName";
        "Südsudanesisches Pfund",
            "displayName-count-one";
        "Südsudanesisches Pfund",
            "displayName-count-other";
        "Südsudanesische Pfund",
            "symbol";
        "SSP",
            "symbol-alt-narrow";
        "£";
    }
    "STD";
    {
        "displayName";
        "São-toméischer Dobra",
            "displayName-count-one";
        "São-toméischer Dobra",
            "displayName-count-other";
        "São-toméische Dobra",
            "symbol";
        "STD",
            "symbol-alt-narrow";
        "Db";
    }
    "SUR";
    {
        "displayName";
        "Sowjetischer Rubel",
            "displayName-count-one";
        "Sowjetische Rubel",
            "displayName-count-other";
        "Sowjetische Rubel",
            "symbol";
    }

```

```

        "SUR";
    }
    "SVC";
    {
        "displayName";
        "El Salvador Colon",
            "displayName-count-one";
        "El Salvador-Colon",
            "displayName-count-other";
        "El Salvador-Colon",
            "symbol";
        "SVC";
    }
    "SYP";
    {
        "displayName";
        "Syrisches Pfund",
            "displayName-count-one";
        "Syrisches Pfund",
            "displayName-count-other";
        "Syrische Pfund",
            "symbol";
        "SYP",
            "symbol-alt-narrow";
        "SYP";
    }
    "SZL";
    {
        "displayName";
        "Swasiländischer Lilangeni",
            "displayName-count-one";
        "Swasiländischer Lilangeni",
            "displayName-count-other";
        "Swasiländische Emalangeni",
            "symbol";
        "SZL";
    }
    "THB";
    {
        "displayName";
        "Thailändischer Baht",
            "displayName-count-one";
        "Thailändischer Baht",
            "displayName-count-other";
        "Thailändische Baht",
            "symbol";
        "฿",
            "symbol-alt-narrow";
        "฿";
    }
    "TJR";
    {
        "displayName";
        "Tadschikistan Rubel",
            "displayName-count-one";
        "Tadschikistan-Rubel",
            "displayName-count-other";
    }

```

```

        "Tadschikistan-Rubel",
        "symbol";
        "TJR";
    }
    "TJS";
    {
        "displayName";
        "Tadschikistan-Somoni",
        "displayName-count-one";
        "Tadschikistan-Somoni",
        "displayName-count-other";
        "Tadschikistan-Somoni",
        "symbol";
        "TJS";
    }
    "TMM";
    {
        "displayName";
        "Turkmenistan-Manat (1993-2009)",
        "displayName-count-one";
        "Turkmenistan-Manat (1993-2009)",
        "displayName-count-other";
        "Turkmenistan-Manat (1993-2009)",
        "symbol";
        "TMM";
    }
    "TMT";
    {
        "displayName";
        "Turkmenistan-Manat",
        "displayName-count-one";
        "Turkmenistan-Manat",
        "displayName-count-other";
        "Turkmenistan-Manat",
        "symbol";
        "TMT";
    }
    "TND";
    {
        "displayName";
        "Tunesischer Dinar",
        "displayName-count-one";
        "Tunesischer Dinar",
        "displayName-count-other";
        "Tunesische Dinar",
        "symbol";
        "TND";
    }
    "TOP";
    {
        "displayName";
        "Tongaischer Pa'anga",
        "displayName-count-one";
        "Tongaischer Pa'anga",
        "displayName-count-other";
        "Tongaische Pa'anga",
        "symbol";
    }

```

```

        "TOP",
        "symbol-alt-narrow";
        "T$";
    }
    "TPE";
    {
        "displayName";
        "Timor-Escudo",
        "displayName-count-one";
        "Timor-Escudo",
        "displayName-count-other";
        "Timor-Escudo",
        "symbol";
        "TPE";
    }
    "TRL";
    {
        "displayName";
        "Türkische Lira (1922-2005)",
        "displayName-count-one";
        "Türkische Lira (1922-2005)",
        "displayName-count-other";
        "Türkische Lira (1922-2005)",
        "symbol";
        "TRL";
    }
    "TRY";
    {
        "displayName";
        "Türkische Lira",
        "displayName-count-one";
        "Türkische Lira",
        "displayName-count-other";
        "Türkische Lira",
        "symbol";
        "TRY",
        "symbol-alt-narrow";
        "₺",
        "symbol-alt-variant";
        "TL";
    }
    "TTD";
    {
        "displayName";
        "Trinidad und Tobago-Dollar",
        "displayName-count-one";
        "Trinidad und Tobago-Dollar",
        "displayName-count-other";
        "Trinidad und Tobago-Dollar",
        "symbol";
        "TTD",
        "symbol-alt-narrow";
        "$";
    }
    "TWD";
    {
        "displayName";

```

```

        "Neuer Taiwan-Dollar",
        "displayName-count-one";
        "Neuer Taiwan-Dollar",
        "displayName-count-other";
        "Neue Taiwan-Dollar",
        "symbol";
        "NT$",
        "symbol-alt-narrow";
        "NT$";
    }
    "TZS";
    {
        "displayName";
        "Tansania-Schilling",
        "displayName-count-one";
        "Tansania-Schilling",
        "displayName-count-other";
        "Tansania-Schilling",
        "symbol";
        "TZS";
    }
    "UAH";
    {
        "displayName";
        "Ukrainische Hrywnja",
        "displayName-count-one";
        "Ukrainische Hrywnja",
        "displayName-count-other";
        "Ukrainische Hrywen",
        "symbol";
        "UAH",
        "symbol-alt-narrow";
        "₴";
    }
    "UAK";
    {
        "displayName";
        "Ukrainischer Karbovanetz",
        "displayName-count-one";
        "Ukrainische Karbovanetz",
        "displayName-count-other";
        "Ukrainische Karbovanetz",
        "symbol";
        "UAK";
    }
    "UGS";
    {
        "displayName";
        "Uganda-Schilling (1966-1987)",
        "displayName-count-one";
        "Uganda-Schilling (1966-1987)",
        "displayName-count-other";
        "Uganda-Schilling (1966-1987)",
        "symbol";
        "UGS";
    }
    "UGX";

```

```

        {
            "displayName";
            "Uganda-Schilling",
            "displayName-count-one";
            "Uganda-Schilling",
            "displayName-count-other";
            "Uganda-Schilling",
            "symbol";
            "UGX";
        }
        "USD";
        {
            "displayName";
            "US-Dollar",
            "displayName-count-one";
            "US-Dollar",
            "displayName-count-other";
            "US-Dollar",
            "symbol";
            "$",
            "symbol-alt-narrow";
            "$";
        }
        "USN";
        {
            "displayName";
            "US Dollar (Nächster Tag)",
            "displayName-count-one";
            "US-Dollar (Nächster Tag)",
            "displayName-count-other";
            "US-Dollar (Nächster Tag)",
            "symbol";
            "USN";
        }
        "USS";
        {
            "displayName";
            "US Dollar (Gleicher Tag)",
            "displayName-count-one";
            "US-Dollar (Gleicher Tag)",
            "displayName-count-other";
            "US-Dollar (Gleicher Tag)",
            "symbol";
            "USS";
        }
        "UYI";
        {
            "displayName";
            "Uruguayischer Peso (Indexierte Rechnungseinheiten)",
            "displayName-count-one";
            "Uruguayischer Peso (Indexierte Rechnungseinheiten)",
            "displayName-count-other";
            "Uruguayische Pesos (Indexierte Rechnungseinheiten)",
            "symbol";
            "UYI";
        }
        "UYP";

```

```

    {
      "displayName";
      "Uruguayischer Peso (1975-1993)",
      "displayName-count-one";
      "Uruguayischer Peso (1975-1993)",
      "displayName-count-other";
      "Uruguayische Pesos (1975-1993)",
      "symbol";
      "UYP";
    }
    "UYU";
    {
      "displayName";
      "Uruguayischer Peso",
      "displayName-count-one";
      "Uruguayischer Peso",
      "displayName-count-other";
      "Uruguayische Pesos",
      "symbol";
      "UYU",
      "symbol-alt-narrow";
      "$";
    }
    "UZS";
    {
      "displayName";
      "Usbekistan-Sum",
      "displayName-count-one";
      "Usbekistan-Sum",
      "displayName-count-other";
      "Usbekistan-Sum",
      "symbol";
      "UZS";
    }
    "VEB";
    {
      "displayName";
      "Venezolanischer Bolívar (1871-2008)",
      "displayName-count-one";
      "Venezolanischer Bolívar (1871-2008)",
      "displayName-count-other";
      "Venezolanische Bolívares (1871-2008)",
      "symbol";
      "VEB";
    }
    "VEF";
    {
      "displayName";
      "Venezolanischer Bolívar",
      "displayName-count-one";
      "Venezolanischer Bolívar",
      "displayName-count-other";
      "Venezolanische Bolívares",
      "symbol";
      "VEF",
      "symbol-alt-narrow";
      "Bs";
    }

```

```

    }
    "VND";
    {
        "displayName";
        "Vietnamesischer Dong",
        "displayName-count-one";
        "Vietnamesischer Dong",
        "displayName-count-other";
        "Vietnamesische Dong",
        "symbol";
        "₫",
        "symbol-alt-narrow";
        "₫";
    }
    "VNN";
    {
        "displayName";
        "Vietnamesischer Dong (1978-1985) ",
        "displayName-count-one";
        "Vietnamesischer Dong (1978-1985) ",
        "displayName-count-other";
        "Vietnamesische Dong (1978-1985) ",
        "symbol";
        "VNN";
    }
    "VUV";
    {
        "displayName";
        "Vanuatu-Vatu",
        "displayName-count-one";
        "Vanuatu-Vatu",
        "displayName-count-other";
        "Vanuatu-Vatu",
        "symbol";
        "VUV";
    }
    "WST";
    {
        "displayName";
        "Samoanischer Tala",
        "displayName-count-one";
        "Samoanischer Tala",
        "displayName-count-other";
        "Samoanische Tala",
        "symbol";
        "WST";
    }
    "XAF";
    {
        "displayName";
        "CFA-Franc (BEAC) ",
        "displayName-count-one";
        "CFA-Franc (BEAC) ",
        "displayName-count-other";
        "CFA-Franc (BEAC) ",
        "symbol";
        "FCFA";
    }

```



```

    }
    "XAG";
    {
        "displayName";
        "Unze Silber",
        "displayName-count-one";
        "Unze Silber",
        "displayName-count-other";
        "Unzen Silber",
        "symbol";
        "XAG";
    }
    "XAU";
    {
        "displayName";
        "Unze Gold",
        "displayName-count-one";
        "Unze Gold",
        "displayName-count-other";
        "Unzen Gold",
        "symbol";
        "XAU";
    }
    "XBA";
    {
        "displayName";
        "Europäische Rechnungseinheit",
        "displayName-count-one";
        "Europäische Rechnungseinheiten",
        "displayName-count-other";
        "Europäische Rechnungseinheiten",
        "symbol";
        "XBA";
    }
    "XBB";
    {
        "displayName";
        "Europäische Währungseinheit (XBB)",
        "displayName-count-one";
        "Europäische Währungseinheiten (XBB)",
        "displayName-count-other";
        "Europäische Währungseinheiten (XBB)",
        "symbol";
        "XBB";
    }
    "XBC";
    {
        "displayName";
        "Europäische Rechnungseinheit (XBC)",
        "displayName-count-one";
        "Europäische Rechnungseinheiten (XBC)",
        "displayName-count-other";
        "Europäische Rechnungseinheiten (XBC)",
        "symbol";
        "XBC";
    }
    "XBD";

```

```

    {
      "displayName";
      "Europäische Rechnungseinheit (XBD)",
        "displayName-count-one";
      "Europäische Rechnungseinheiten (XBD)",
        "displayName-count-other";
      "Europäische Rechnungseinheiten (XBD)",
        "symbol";
      "XBD";
    }
    "XCD";
    {
      "displayName";
      "Ostkaribischer Dollar",
        "displayName-count-one";
      "Ostkaribischer Dollar",
        "displayName-count-other";
      "Ostkaribische Dollar",
        "symbol";
      "EC$",
        "symbol-alt-narrow";
      "$";
    }
    "XDR";
    {
      "displayName";
      "Sonderziehungsrechte",
        "displayName-count-one";
      "Sonderziehungsrechte",
        "displayName-count-other";
      "Sonderziehungsrechte",
        "symbol";
      "XDR";
    }
    "XEU";
    {
      "displayName";
      "Europäische Währungseinheit (XEU)",
        "displayName-count-one";
      "Europäische Währungseinheiten (XEU)",
        "displayName-count-other";
      "Europäische Währungseinheiten (XEU)",
        "symbol";
      "XEU";
    }
    "XFO";
    {
      "displayName";
      "Französischer Gold-Franc",
        "displayName-count-one";
      "Französische Gold-Franc",
        "displayName-count-other";
      "Französische Gold-Franc",
        "symbol";
      "XFO";
    }
    "XFU";

```

```

        {
            "displayName";
            "Französischer UIC-Franc",
            "displayName-count-one";
            "Französische UIC-Franc",
            "displayName-count-other";
            "Französische UIC-Franc",
            "symbol";
            "XFU";
        }
        "XOF";
        {
            "displayName";
            "CFA-Franc (BCEAO)",
            "displayName-count-one";
            "CFA-Franc (BCEAO)",
            "displayName-count-other";
            "CFA-Francs (BCEAO)",
            "symbol";
            "CFA";
        }
        "XPD";
        {
            "displayName";
            "Unze Palladium",
            "displayName-count-one";
            "Unze Palladium",
            "displayName-count-other";
            "Unzen Palladium",
            "symbol";
            "XPD";
        }
        "XPF";
        {
            "displayName";
            "CFP-Franc",
            "displayName-count-one";
            "CFP-Franc",
            "displayName-count-other";
            "CFP-Franc",
            "symbol";
            "CFPF";
        }
        "XPT";
        {
            "displayName";
            "Unze Platin",
            "displayName-count-one";
            "Unze Platin",
            "displayName-count-other";
            "Unzen Platin",
            "symbol";
            "XPT";
        }
        "XRE";
        {
            "displayName";

```

```

        "RINET Funds",
        "displayName-count-one";
        "RINET Funds",
        "displayName-count-other";
        "RINET Funds",
        "symbol";
        "XRE";
    }
    "XSU";
    {
        "displayName";
        "SUCRE",
        "displayName-count-one";
        "SUCRE",
        "displayName-count-other";
        "SUCRE",
        "symbol";
        "XSU";
    }
    "XTS";
    {
        "displayName";
        "Testwährung",
        "displayName-count-one";
        "Testwährung",
        "displayName-count-other";
        "Testwährung",
        "symbol";
        "XTS";
    }
    "XUA";
    {
        "displayName";
        "Rechnungseinheit der AfEB",
        "displayName-count-one";
        "Rechnungseinheit der AfEB",
        "displayName-count-other";
        "Rechnungseinheiten der AfEB",
        "symbol";
        "XUA";
    }
    "XXX";
    {
        "displayName";
        "Unbekannte Währung",
        "displayName-count-one";
        "(unbekannte Währung)",
        "displayName-count-other";
        "(unbekannte Währung)",
        "symbol";
        "XXX";
    }
    "YDD";
    {
        "displayName";
        "Jemen-Dinar",
        "displayName-count-one";

```

```

        "Jemen-Dinar",
        "displayName-count-other";
        "Jemen-Dinar",
        "symbol";
        "YDD";
    }
    "YER";
    {
        "displayName";
        "Jemen-Rial",
        "displayName-count-one";
        "Jemen-Rial",
        "displayName-count-other";
        "Jemen-Rial",
        "symbol";
        "YER";
    }
    "YUD";
    {
        "displayName";
        "Jugoslawischer Dinar (1966-1990)",
        "displayName-count-one";
        "Jugoslawischer Dinar (1966-1990)",
        "displayName-count-other";
        "Jugoslawische Dinar (1966-1990)",
        "symbol";
        "YUD";
    }
    "YUM";
    {
        "displayName";
        "Jugoslawischer Neuer Dinar (1994-2002)",
        "displayName-count-one";
        "Jugoslawischer Neuer Dinar (1994-2002)",
        "displayName-count-other";
        "Jugoslawische Neue Dinar (1994-2002)",
        "symbol";
        "YUM";
    }
    "YUN";
    {
        "displayName";
        "Jugoslawischer Dinar (konvertibel)",
        "displayName-count-one";
        "Jugoslawische Dinar (konvertibel)",
        "displayName-count-other";
        "Jugoslawische Dinar (konvertibel)",
        "symbol";
        "YUN";
    }
    "YUR";
    {
        "displayName";
        "Jugoslawischer reformierter Dinar (1992-1993)",
        "displayName-count-one";
        "Jugoslawischer reformierter Dinar (1992-1993)",
        "displayName-count-other";
    }

```

```

        "Jugoslawische reformierte Dinar (1992-1993)",
        "symbol";
        "YUR";
    }
    "ZAL";
    {
        "displayName";
        "Südafrikanischer Rand (Finanz)",
        "displayName-count-one";
        "Südafrikanischer Rand (Finanz)",
        "displayName-count-other";
        "Südafrikanischer Rand (Finanz)",
        "symbol";
        "ZAL";
    }
    "ZAR";
    {
        "displayName";
        "Südafrikanischer Rand",
        "displayName-count-one";
        "Südafrikanischer Rand",
        "displayName-count-other";
        "Südafrikanische Rand",
        "symbol";
        "ZAR",
        "symbol-alt-narrow";
        "R";
    }
    "ZMK";
    {
        "displayName";
        "Kwacha (1968-2012)",
        "displayName-count-one";
        "Kwacha (1968-2012)",
        "displayName-count-other";
        "Kwacha (1968-2012)",
        "symbol";
        "ZMK";
    }
    "ZMW";
    {
        "displayName";
        "Kwacha",
        "displayName-count-one";
        "Kwacha",
        "displayName-count-other";
        "Kwacha",
        "symbol";
        "ZMW",
        "symbol-alt-narrow";
        "K";
    }
    "ZRN";
    {
        "displayName";
        "Zaire-Neuer Zaïre (1993-1998)",
        "displayName-count-one";

```

```

        "Zaire-Neuer Zaïre (1993-1998)",
            "displayName-count-other";
        "Zaire-Neue Zaïre (1993-1998)",
            "symbol";
        "ZRN";
    }
    "ZRZ";
    {
        "displayName";
        "Zaire-Zaïre (1971-1993)",
            "displayName-count-one";
        "Zaire-Zaïre (1971-1993)",
            "displayName-count-other";
        "Zaire-Zaïre (1971-1993)",
            "symbol";
        "ZRZ";
    }
    "ZWD";
    {
        "displayName";
        "Simbabwe-Dollar (1980-2008)",
            "displayName-count-one";
        "Simbabwe-Dollar (1980-2008)",
            "displayName-count-other";
        "Simbabwe-Dollar (1980-2008)",
            "symbol";
        "ZWD";
    }
    "ZWL";
    {
        "displayName";
        "Simbabwe-Dollar (2009)",
            "displayName-count-one";
        "Simbabwe-Dollar (2009)",
            "displayName-count-other";
        "Simbabwe-Dollar (2009)",
            "symbol";
        "ZWL";
    }
    "ZWR";
    {
        "displayName";
        "Simbabwe-Dollar (2008)",
            "displayName-count-one";
        "Simbabwe-Dollar (2008)",
            "displayName-count-other";
        "Simbabwe-Dollar (2008)",
            "symbol";
        "ZWR";
    }
}
}
}
}
}

```

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
//Load the L10n, loadCldr from ej2-base
import { loadCldr, L10n } from '@syncfusion/ej2-base';
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
//load the CLDR data files.
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as deTimeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, deTimeZoneNames);
L10n.load({
    'de': {
        'daterangepicker': {
            applyText: 'Sich bewerben',
            cancelText: 'Stornieren',
            customRange: 'benutzerdefinierten Bereich',
            days: 'Tage',
            endLabel: 'Wählen Sie Enddatum',
            placeholder: 'Wählen Sie einen Bereich aus',
            selectedDays: 'Ausgewählte Tage',
            startLabel: 'Wählen Sie Startdatum'
        }
    }
});
//import the daterangepicker component
class App extends React.Component {
    render() {
        return <DateRangePickerComponent id="daterangepicker" locale='de'/>;
    }
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
//Load the L10n, loadCldr from ej2-base
import { loadCldr, L10n } from '@syncfusion/ej2-base';
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
//load the CLDR data files.
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as deTimeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, deTimeZoneNames);
L10n.load({
    'de': {
        'daterangepicker': {
            applyText: 'Sich bewerben',
            cancelText: 'Stornieren',

```



```

        customRange: 'benutzerdefinierten Bereich',
        days: 'Tage',
        endLabel: 'Wählen Sie Enddatum',
        placeholder: 'Wählen Sie einen Bereich aus',
        selectedDays: 'Ausgewählte Tage',
        startLabel: 'Wählen Sie Startdatum'
    }
}
});
//import the daterangepicker component
class App extends React.Component<{}, {}> {
    render() {
        return <DateRangePickerComponent id="daterangepicker" locale='de' />
    }
};
ReactDOM.render(<App />, document.getElementById('element'));

```

NUMBERINGSYSTEMS.JSON

```

{
  "supplemental": {
    "version": {
      "_number": "$Revision: 12732 $",
      "_unicodeVersion": "9.0.0",
      "_cldrVersion": "31"
    },
    "numberingSystems": {
      "adlm": {
        "_digits": "ᲠᲡᲢᲣᲤᲥᲦᲧᲨᲩ",
        "_type": "numeric"
      },
      "ahom": {
        "_digits": "ᩌᩍᩎᩏᩐᩑᩒᩓᩔᩕ",
        "_type": "numeric"
      },
      "arab": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "arabext": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "armn": {
        "_rules": "armenian-upper",
        "_type": "algorithmic"
      },
      "armnlow": {
        "_rules": "armenian-lower",
        "_type": "algorithmic"
      },
      "bali": {
        "_digits": "ᬀᬁᬂᬃᬄᬅᬆᬇᬈᬉ",
        "_type": "numeric"
      },
      "beng": {

```

```

    "_digits": "᠐᠑᠒᠐8᠘᠖9᠜ᠨ",
    "_type": "numeric"
  },
  "bhks": {
    "_digits": "᠐᠐᠐᠐᠐᠐᠐᠐᠐᠐",
    "_type": "numeric"
  },
  "brah": {
    "_digits": "᠐᠕᠒᠔᠕᠑᠓᠙",
    "_type": "numeric"
  },
  "cakm": {
    "_digits": "᠐ᠪᠶ᠋ᠭᠢᠨᠠᠨᠢᠨᠠᠨ",
    "_type": "numeric"
  },
  "cham": {
    "_digits": "᠐᠐᠐᠐᠐᠐᠐᠐᠐᠐",
    "_type": "numeric"
  },
  "cyr1": {
    "_rules": "cyrillic-lower",
    "_type": "algorithmic"
  },
  "deva": {
    "_digits": "᠐᠑᠒᠔᠕᠖ᠭᠨᠠᠨ",
    "_type": "numeric"
  },
  "ethi": {
    "_rules": "ethiopic",
    "_type": "algorithmic"
  },
  "fullwide": {
    "_digits": "0 1 2 3 4 5 6 7 8 9",
    "_type": "numeric"
  },
  "geor": {
    "_rules": "georgian",
    "_type": "algorithmic"
  },
  "grek": {
    "_rules": "greek-upper",
    "_type": "algorithmic"
  },
  "greklow": {
    "_rules": "greek-lower",
    "_type": "algorithmic"
  },
  "gujr": {
    "_digits": "᠐᠑᠒᠔᠕᠖ᠭᠨᠠᠨ",
    "_type": "numeric"
  },
  "guru": {
    "_digits": "᠐᠑᠒᠔᠕᠖ᠭᠨᠠᠨ",
    "_type": "numeric"
  },

```

```

"hanidays": {
  "_rules": "zh/SpelloutRules/spellout-numbering-days",
  "_type": "algorithmic"
},
"hanidec": {
  "_digits": "〇一二三四五六七八九",
  "_type": "numeric"
},
"hans": {
  "_rules": "zh/SpelloutRules/spellout-cardinal",
  "_type": "algorithmic"
},
"hansfin": {
  "_rules": "zh/SpelloutRules/spellout-cardinal-financial",
  "_type": "algorithmic"
},
"hant": {
  "_rules": "zh_Hant/SpelloutRules/spellout-cardinal",
  "_type": "algorithmic"
},
"hantfin": {
  "_rules": "zh_Hant/SpelloutRules/spellout-cardinal-financial",
  "_type": "algorithmic"
},
"hebr": {
  "_rules": "hebrew",
  "_type": "algorithmic"
},
"hmng": {
  "_digits": "᠐᠑᠒᠓᠐ᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨ",
  "_type": "numeric"
},
"java": {
  "_digits": "᠐ᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨ",
  "_type": "numeric"
},
"jpan": {
  "_rules": "ja/SpelloutRules/spellout-cardinal",
  "_type": "algorithmic"
},
"jpanfin": {
  "_rules": "ja/SpelloutRules/spellout-cardinal-financial",
  "_type": "algorithmic"
},
"kali": {
  "_digits": "᠐ᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨᠠᠨ",
  "_type": "numeric"
},
"khmr": {
  "_digits": "០១២៣៤៥៦៧៨៩",
  "_type": "numeric"
},
"knda": {
  "_digits": "೦೧೨೩೪೫೬೭೮೯",
  "_type": "numeric"
}

```

```

    },
    "lana": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    },
    "lanatham": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    },
    "laoo": {
      "_digits": "໐໑໒໓໔໕໖໗໘໑",
      "_type": "numeric"
    },
    },
    "latn": {
      "_digits": "0123456789",
      "_type": "numeric"
    },
    },
    "lepc": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    },
    "limb": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    },
    "mathbold": {
      "_digits": "0123456789",
      "_type": "numeric"
    },
    },
    "mathdbl": {
      "_digits": "0123456789",
      "_type": "numeric"
    },
    },
    "mathmono": {
      "_digits": "0123456789",
      "_type": "numeric"
    },
    },
    "mathsanb": {
      "_digits": "0123456789",
      "_type": "numeric"
    },
    },
    "mathsans": {
      "_digits": "0123456789",
      "_type": "numeric"
    },
    },
    "mlym": {
      "_digits": "൦൧൨൩൪൫൬൭൮൯",
      "_type": "numeric"
    },
    },
    "modi": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    },
    "mong": {
      "_digits": "᠐᠑᠒᠓᠐ᠠᠨᠨᠠᠭᠤᠯᠤᠰ",
      "_type": "numeric"
    }
  },

```

```

    },
    "mroo": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    "mtei": {
      "_digits": "၀၀၀၀၀၀၀၀၀၀၀၀",
      "_type": "numeric"
    },
    "mymr": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "mymrshan": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "mymrtlng": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "newa": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    "nkoo": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "olck": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "orya": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "osma": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "roman": {
      "_rules": "roman-upper",
      "_type": "algorithmic"
    },
    "romanlow": {
      "_rules": "roman-lower",
      "_type": "algorithmic"
    },
    "saur": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    "shrd": {

```

```

    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "sind": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "sinh": {
    "_digits": "𑌰𑌱𑌲𑌳𑌴𑌵𑌶𑌷𑌸𑌹𑌺𑌻𑌼𑌽𑌾𑌿",
    "_type": "numeric"
  },
  "sora": {
    "_digits": "ᱵᱚᱠᱟᱨᱚᱸᱰᱟᱦᱚᱸᱰᱚᱨ",
    "_type": "numeric"
  },
  "sund": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "takr": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "talu": {
    "_digits": "ᱠᱚᱱᱚᱴᱚᱨᱚᱸᱰᱟᱦᱚᱸᱰᱚᱨ",
    "_type": "numeric"
  },
  "taml": {
    "_rules": "tamil",
    "_type": "algorithmic"
  },
  "tamldec": {
    "_digits": "௦௧௨௩௪௫௬௭௮௯",
    "_type": "numeric"
  },
  "telu": {
    "_digits": "౦౧౨౩౪౫౬౭౮౯",
    "_type": "numeric"
  },
  "thai": {
    "_digits": "๐๑๒๓๔๕๖๗๘๙",
    "_type": "numeric"
  },
  "tibt": {
    "_digits": "༠༡༢༣༤༥༦༧༨༩",
    "_type": "numeric"
  },
  "tirh": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "vaih": {
    "_digits": "ᱠᱚᱱᱚᱴᱚᱨᱚᱸᱰᱟᱦᱚᱸᱰᱚᱨ",
    "_type": "numeric"
  },

```

```

    "wara": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    }
  }
}

```

NUMBERINGSYSTEMS.JSX

```

{
  "supplemental";
  {
    "version";
    {
      "_number";
      "$Revision: 12732 $",
      "_unicodeVersion";
      "9.0.0",
      "_cldrVersion";
      "31";
    }
    "numberingSystems";
    {
      "adlm";
      {
        "_digits";
        "୧୨୩୪୫୬୭୮୯୦",
        "_type";
        "numeric";
      }
      "ahom";
      {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
      }
      "arab";
      {
        "_digits";
        "٠١٢٣٤٥٦٧٨٩",
        "_type";
        "numeric";
      }
      "arabext";
      {
        "_digits";
        "٠١٢٣٤٥٦٧٨٩",
        "_type";
        "numeric";
      }
      "armn";
      {
        "_rules";
        "armenian-upper",

```

```

        "_type";
        "algorithmic";
    }
    "armnlow";
    {
        "_rules";
        "armenian-lower",
        "_type";
        "algorithmic";
    }
    "bali";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "beng";
    {
        "_digits";
        "০১২৩৪৫৬৭৮৯",
        "_type";
        "numeric";
    }
    "bhks";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "brah";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "cakm";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "cham";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "cyrl";
    {
        "_rules";
        "cyrillic-lower",

```



```

        "_type";
        "algorithmic";
    }
    "deva";
    {
        "_digits";
        "୦୧୨୩୪୫୬୭୮୯",
        "_type";
        "numeric";
    }
    "ethi";
    {
        "_rules";
        "ethiopic",
        "_type";
        "algorithmic";
    }
    "fullwide";
    {
        "_digits";
        "0 1 2 3 4 5 6 7 8 9",
        "_type";
        "numeric";
    }
    "geor";
    {
        "_rules";
        "georgian",
        "_type";
        "algorithmic";
    }
    "grek";
    {
        "_rules";
        "greek-upper",
        "_type";
        "algorithmic";
    }
    "greklow";
    {
        "_rules";
        "greek-lower",
        "_type";
        "algorithmic";
    }
    "gujr";
    {
        "_digits";
        "୦୧୨୩୪୫୬୭୮୯",
        "_type";
        "numeric";
    }
    "guru";
    {
        "_digits";
        "୦୧୨୩୪୫୬୭୮୯",

```

```

        "_type";
        "numeric";
    }
    "hanidays";
    {
        "_rules";
        "zh/SpelloutRules/spellout-numbering-days",
        "_type";
        "algorithmic";
    }
    "hanidec";
    {
        "_digits";
        "〇一二三四五六七八九",
        "_type";
        "numeric";
    }
    "hans";
    {
        "_rules";
        "zh/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "hansfin";
    {
        "_rules";
        "zh/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "hant";
    {
        "_rules";
        "zh_Hant/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "hantfin";
    {
        "_rules";
        "zh_Hant/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "hebr";
    {
        "_rules";
        "hebrew",
        "_type";
        "algorithmic";
    }
    "hmng";
    {
        "_digits";
        "□□□□□□□□□□",

```

```

        "_type";
        "numeric";
    }
    "java";
    {
        "_digits";
        "௦௧௨௩௪௫௬௭௮௯",
        "_type";
        "numeric";
    }
    "jpan";
    {
        "_rules";
        "ja/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "jpanfin";
    {
        "_rules";
        "ja/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "kali";
    {
        "_digits";
        "௦௧௨௩௪௫௬௭௮௯",
        "_type";
        "numeric";
    }
    "khmr";
    {
        "_digits";
        "០១២៣៤៥៦៧៨៩",
        "_type";
        "numeric";
    }
    "knda";
    {
        "_digits";
        "೦೧೨೩೪೫೬೭೮೯",
        "_type";
        "numeric";
    }
    "lana";
    {
        "_digits";
        "௦௧௨௩௪௫௬௭௮௯",
        "_type";
        "numeric";
    }
    "lanatham";
    {
        "_digits";

```

```

        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "laoo";
    {
        "_digits";
        "໐໑໒໓໔໕໖໗໘໑",
        "_type";
        "numeric";
    }
    "latn";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "lepc";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "limb";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "mathbold";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathdbl";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathmono";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathsanb";
    {
        "_digits";

```

```

        "0123456789",
        "_type";
        "numeric";
    }
    "mathsans";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mlym";
    {
        "_digits";
        "ഘറനർത്ഥനവുൻ",
        "_type";
        "numeric";
    }
    "modi";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "mong";
    {
        "_digits";
        "᠎ᠠᠨᠮᠣᠩᠭ᠎ᠠ",
        "_type";
        "numeric";
    }
    "mroo";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "mtei";
    {
        "_digits";
        "ၵၵၵၵၵၵၵၵၵၵ",
        "_type";
        "numeric";
    }
    "mymr";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "mymrshan";
    {

```

```

        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "myrtl";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "newa";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "nkoo";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "olck";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "orya";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "osma";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "roman";
    {
        "_rules";
        "roman-upper",
        "_type";
        "algorithmic";
    }

```

[illegible]

```

"taluk";
{
  "_digits";
  "0123456789",
  "_type";
  "numeric";
}
"tam1";
{
  "_rules";
  "tam1",
  "_type";
  "algorithmic";
}
"tam1dec";
{
  "_digits";
  "0௧௨௩௪௫௬௭௮௯",
  "_type";
  "numeric";
}
"telu";
{
  "_digits";
  "0౧౨౩౪౫౬౭౮౯",
  "_type";
  "numeric";
}
"thai";
{
  "_digits";
  "๐๑๒๓๔๕๖๗๘๙",
  "_type";
  "numeric";
}
"tib1";
{
  "_digits";
  "0123456789",
  "_type";
  "numeric";
}
"tirh";
{
  "_digits";
  "0123456789",
  "_type";
  "numeric";
}
"vaii";
{
  "_digits";
  "0123456789",
  "_type";
  "numeric";
}

```



```

    }
    "wara";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
}
}

```

NUMBERS.JSON

```

{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31"
        },
        "language": "de"
      },
      "numbers": {
        "defaultNumberingSystem": "latn",
        "otherNumberingSystems": {
          "native": "latn"
        },
        "minimumGroupingDigits": "1",
        "symbols-numberSystem-latn": {
          "decimal": ",",
          "group": ".",
          "list": ";",
          "percentSign": "%",
          "plusSign": "+",
          "minusSign": "-",
          "exponential": "E",
          "superscriptingExponent": "°",
          "perMille": "‰",
          "infinity": "∞",
          "nan": "NaN",
          "timeSeparator": ":"
        },
        "decimalFormats-numberSystem-latn": {
          "standard": "#,##0.###",
          "long": {
            "decimalFormat": {
              "1000-count-one": "0 Tausend",
              "1000-count-other": "0 Tausend",
              "10000-count-one": "00 Tausend",
              "10000-count-other": "00 Tausend",
              "100000-count-one": "000 Tausend",
              "100000-count-other": "000 Tausend",
              "1000000-count-one": "0 Million",
              "1000000-count-other": "0 Millionen",
            }
          }
        }
      }
    }
  }
}

```

```

        "10000000-count-one": "00 Millionen",
        "10000000-count-other": "00 Millionen",
        "100000000-count-one": "000 Millionen",
        "100000000-count-other": "000 Millionen",
        "1000000000-count-one": "0 Milliarden",
        "1000000000-count-other": "0 Milliarden",
        "10000000000-count-one": "00 Milliarden",
        "10000000000-count-other": "00 Milliarden",
        "100000000000-count-one": "000 Milliarden",
        "100000000000-count-other": "000 Milliarden",
        "1000000000000-count-one": "0 Billion",
        "1000000000000-count-other": "0 Billionen",
        "10000000000000-count-one": "00 Billionen",
        "10000000000000-count-other": "00 Billionen",
        "100000000000000-count-one": "000 Billionen",
        "100000000000000-count-other": "000 Billionen"
    },
    },
    "short": {
        "decimalFormat": {
            "1000-count-one": "0",
            "1000-count-other": "0",
            "10000-count-one": "0",
            "10000-count-other": "0",
            "100000-count-one": "0",
            "100000-count-other": "0",
            "1000000-count-one": "0 Mio'.'",
            "1000000-count-other": "0 Mio'.'",
            "10000000-count-one": "00 Mio'.'",
            "10000000-count-other": "00 Mio'.'",
            "100000000-count-one": "000 Mio'.'",
            "100000000-count-other": "000 Mio'.'",
            "1000000000-count-one": "0 Mrd'.'",
            "1000000000-count-other": "0 Mrd'.'",
            "10000000000-count-one": "00 Mrd'.'",
            "10000000000-count-other": "00 Mrd'.'",
            "100000000000-count-one": "000 Mrd'.'",
            "100000000000-count-other": "000 Mrd'.'",
            "1000000000000-count-one": "0 Bio'.'",
            "1000000000000-count-other": "0 Bio'.'",
            "10000000000000-count-one": "00 Bio'.'",
            "10000000000000-count-other": "00 Bio'.'",
            "100000000000000-count-one": "000 Bio'.'",
            "100000000000000-count-other": "000 Bio'.'"
        }
    }
},
"scientificFormats-numberSystem-latn": {
    "standard": "#E0"
},
"percentFormats-numberSystem-latn": {
    "standard": "#,##0 %"
},
"currencyFormats-numberSystem-latn": {
    "currencySpacing": {
        "beforeCurrency": {
            "currencyMatch": "[:^S:]",

```

```

        "surroundingMatch": "[:digit:]",
        "insertBetween": " "
    },
    "afterCurrency": {
        "currencyMatch": "[:^S:]",
        "surroundingMatch": "[:digit:]",
        "insertBetween": " "
    }
},
"standard": "#,##0.00 ¤",
"accounting": "#,##0.00 ¤",
"short": {
    "standard": {
        "1000-count-one": "0 Tsd'.' ¤",
        "1000-count-other": "0 Tsd'.' ¤",
        "10000-count-one": "00 Tsd'.' ¤",
        "10000-count-other": "00 Tsd'.' ¤",
        "100000-count-one": "000 Tsd'.' ¤",
        "100000-count-other": "000 Tsd'.' ¤",
        "1000000-count-one": "0 Mio'.' ¤",
        "1000000-count-other": "0 Mio'.' ¤",
        "10000000-count-one": "00 Mio'.' ¤",
        "10000000-count-other": "00 Mio'.' ¤",
        "100000000-count-one": "000 Mio'.' ¤",
        "100000000-count-other": "000 Mio'.' ¤",
        "1000000000-count-one": "0 Mrd'.' ¤",
        "1000000000-count-other": "0 Mrd'.' ¤",
        "10000000000-count-one": "00 Mrd'.' ¤",
        "10000000000-count-other": "00 Mrd'.' ¤",
        "100000000000-count-one": "000 Mrd'.' ¤",
        "100000000000-count-other": "000 Mrd'.' ¤",
        "1000000000000-count-one": "0 Bio'.' ¤",
        "1000000000000-count-other": "0 Bio'.' ¤",
        "10000000000000-count-one": "00 Bio'.' ¤",
        "10000000000000-count-other": "00 Bio'.' ¤",
        "100000000000000-count-one": "000 Bio'.' ¤",
        "100000000000000-count-other": "000 Bio'.' ¤"
    }
},
"unitPattern-count-one": "{0} {1}",
"unitPattern-count-other": "{0} {1}"
},
"miscPatterns-numberSystem-latn": {
    "atLeast": "{0}+",
    "range": "{0}-{1}"
},
"minimalPairs": {
    "pluralMinimalPairs": "{0} Tag",
    "pluralMinimalPairs": "{0} Tage",
    "other": "{0}. Abzweigung nach rechts nehmen"
}
}
}
}
}
}

```

NUMBERS.JSX

```
{
  "main";
  {
    "de";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 13259 $",
          "_cldrVersion";
          "31";
        }
        "language";
        "de";
      }
    }
    "numbers";
    {
      "defaultNumberingSystem";
      "latn",
      "otherNumberingSystems";
      {
        "native";
        "latn";
      }
      "minimumGroupingDigits";
      "1",
      "symbols-numberSystem-latn";
      {
        "decimal";
        ",",
        "group";
        ".",
        "list";
        ";",
        "percentSign";
        "%",
        "plusSign";
        "+",
        "minusSign";
        "-",
        "exponential";
        "E",
        "superscriptingExponent";
        ". ",
        "perMille";
        "‰",
        "infinity";
        "∞",
        "nan";
        "NaN",
        "timeSeparator";
        ":";
      }
    }
  }
}
```

```

    "decimalFormats-numberSystem-latn";
    {
        "standard";
        "#,##0.###",
        "long";
        {
            "decimalFormat";
            {
                "1000-count-one";
                "0 Tausend",
                "1000-count-other";
                "0 Tausend",
                "10000-count-one";
                "00 Tausend",
                "10000-count-other";
                "00 Tausend",
                "100000-count-one";
                "000 Tausend",
                "100000-count-other";
                "000 Tausend",
                "1000000-count-one";
                "0 Million",
                "1000000-count-other";
                "0 Millionen",
                "10000000-count-one";
                "00 Millionen",
                "10000000-count-other";
                "00 Millionen",
                "100000000-count-one";
                "000 Millionen",
                "100000000-count-other";
                "000 Millionen",
                "1000000000-count-one";
                "0 Milliarde",
                "1000000000-count-other";
                "0 Milliarden",
                "10000000000-count-one";
                "00 Milliarden",
                "10000000000-count-other";
                "00 Milliarden",
                "100000000000-count-one";
                "000 Milliarden",
                "100000000000-count-other";
                "000 Milliarden",
                "1000000000000-count-one";
                "0 Billion",
                "1000000000000-count-other";
                "0 Billionen",
                "10000000000000-count-one";
                "00 Billionen",
                "10000000000000-count-other";
                "00 Billionen",
                "100000000000000-count-one";
                "000 Billionen",
                "100000000000000-count-other";
                "000 Billionen";
            }
        }
    }

```

```

    }
    "short";
    {
        "decimalFormat";
        {
            "1000-count-one";
            "0",
            "1000-count-other";
            "0",
            "10000-count-one";
            "0",
            "10000-count-other";
            "0",
            "100000-count-one";
            "0",
            "100000-count-other";
            "0",
            "1000000-count-one";
            "0 Mio'.'",
            "1000000-count-other";
            "0 Mio'.'",
            "10000000-count-one";
            "00 Mio'.'",
            "10000000-count-other";
            "00 Mio'.'",
            "100000000-count-one";
            "000 Mio'.'",
            "100000000-count-other";
            "000 Mio'.'",
            "1000000000-count-one";
            "0 Mrd'.'",
            "1000000000-count-other";
            "0 Mrd'.'",
            "10000000000-count-one";
            "00 Mrd'.'",
            "10000000000-count-other";
            "00 Mrd'.'",
            "100000000000-count-one";
            "000 Mrd'.'",
            "100000000000-count-other";
            "000 Mrd'.'",
            "1000000000000-count-one";
            "0 Bio'.'",
            "1000000000000-count-other";
            "0 Bio'.'",
            "10000000000000-count-one";
            "00 Bio'.'",
            "10000000000000-count-other";
            "00 Bio'.'",
            "100000000000000-count-one";
            "000 Bio'.'",
            "100000000000000-count-other";
            "000 Bio'.'";
        }
    }
}
"scientificFormats-numberSystem-latn";

```

```

    {
      "standard";
      "#E0";
    }
    "percentFormats-numberSystem-latn";
    {
      "standard";
      "#,##0 %";
    }
    "currencyFormats-numberSystem-latn";
    {
      "currencySpacing";
      {
        "beforeCurrency";
        {
          "currencyMatch";
          "[:^S:]",
          "surroundingMatch";
          "[:digit:]",
          "insertBetween";
          " ";
        }
        "afterCurrency";
        {
          "currencyMatch";
          "[:^S:]",
          "surroundingMatch";
          "[:digit:]",
          "insertBetween";
          " ";
        }
      }
    }
    "standard";
    "#,##0.00 ¤",
    "accounting";
    "#,##0.00 ¤",
    "short";
    {
      "standard";
      {
        "1000-count-one";
        "0 Tsd'.' ¤",
        "1000-count-other";
        "0 Tsd'.' ¤",
        "10000-count-one";
        "00 Tsd'.' ¤",
        "10000-count-other";
        "00 Tsd'.' ¤",
        "100000-count-one";
        "000 Tsd'.' ¤",
        "100000-count-other";
        "000 Tsd'.' ¤",
        "1000000-count-one";
        "0 Mio'.' ¤",
        "1000000-count-other";
        "0 Mio'.' ¤",
        "10000000-count-one";
      }
    }
  }

```

```

        "00 Mio'.' x",
        "10000000-count-other";
        "00 Mio'.' x",
        "100000000-count-one";
        "000 Mio'.' x",
        "100000000-count-other";
        "000 Mio'.' x",
        "1000000000-count-one";
        "0 Mrd'.' x",
        "1000000000-count-other";
        "0 Mrd'.' x",
        "10000000000-count-one";
        "00 Mrd'.' x",
        "10000000000-count-other";
        "00 Mrd'.' x",
        "100000000000-count-one";
        "000 Mrd'.' x",
        "100000000000-count-other";
        "000 Mrd'.' x",
        "1000000000000-count-one";
        "0 Bio'.' x",
        "1000000000000-count-other";
        "0 Bio'.' x",
        "10000000000000-count-one";
        "00 Bio'.' x",
        "10000000000000-count-other";
        "00 Bio'.' x",
        "100000000000000-count-one";
        "000 Bio'.' x",
        "100000000000000-count-other";
        "000 Bio'.' x";
    }
}
"unitPattern-count-one";
"{0} {1}",
    "unitPattern-count-other";
"{0} {1}";
}
"miscPatterns-numberSystem-latn";
{
    "atLeast";
    "{0}+",
        "range";
    "{0}-{1}";
}
"minimalPairs";
{
    "pluralMinimalPairs";
    "{0} Tag",
        "pluralMinimalPairs";
    "{0} Tage",
        "other";
    "{0}. Abzweigung nach rechts nehmen";
}
}
}
}

```



```
}
```

TIMEZONENAMES.JSON

```
{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 12879 $",
          "_cldrVersion": "30.0.3"
        },
        "language": "de"
      },
      "dates": {
        "timeZoneNames": {
          "hourFormat": "+HH:mm;-HH:mm",
          "gmtFormat": "GMT{0}",
          "gmtZeroFormat": "GMT",
          "regionFormat": "{0} Zeit",
          "regionFormat-type-daylight": "{0} Sommerzeit",
          "regionFormat-type-standard": "{0} Normalzeit",
          "fallbackFormat": "{1} ({0})",
          "zone": {
            "America": {
              "Adak": {
                "exemplarCity": "Adak"
              },
              "Anchorage": {
                "exemplarCity": "Anchorage"
              },
              "Anguilla": {
                "exemplarCity": "Anguilla"
              },
              "Antigua": {
                "exemplarCity": "Antigua"
              },
              "Araguaina": {
                "exemplarCity": "Araguaina"
              },
              "Argentina": {
                "Rio_Gallegos": {
                  "exemplarCity": "Rio Gallegos"
                },
                "San_Juan": {
                  "exemplarCity": "San Juan"
                },
                "Ushuaia": {
                  "exemplarCity": "Ushuaia"
                },
                "La_Rioja": {
                  "exemplarCity": "La Rioja"
                },
                "San_Luis": {
                  "exemplarCity": "San Luis"
                }
              }
            }
          }
        }
      }
    }
  }
}
```

```
    "Salta": {
      "exemplarCity": "Salta"
    },
    "Tucuman": {
      "exemplarCity": "Tucuman"
    }
  },
  "Aruba": {
    "exemplarCity": "Aruba"
  },
  "Asuncion": {
    "exemplarCity": "Asunción"
  },
  "Bahia": {
    "exemplarCity": "Bahia"
  },
  "Bahia_Banderas": {
    "exemplarCity": "Bahia Banderas"
  },
  "Barbados": {
    "exemplarCity": "Barbados"
  },
  "Belem": {
    "exemplarCity": "Belem"
  },
  "Belize": {
    "exemplarCity": "Belize"
  },
  "Blanc-Sablon": {
    "exemplarCity": "Blanc-Sablon"
  },
  "Boa_Vista": {
    "exemplarCity": "Boa Vista"
  },
  "Bogota": {
    "exemplarCity": "Bogotá"
  },
  "Boise": {
    "exemplarCity": "Boise"
  },
  "Buenos_Aires": {
    "exemplarCity": "Buenos Aires"
  },
  "Cambridge_Bay": {
    "exemplarCity": "Cambridge Bay"
  },
  "Campo_Grande": {
    "exemplarCity": "Campo Grande"
  },
  "Cancun": {
    "exemplarCity": "Cancún"
  },
  "Caracas": {
    "exemplarCity": "Caracas"
  },
  "Catamarca": {
    "exemplarCity": "Catamarca"
  }
```

```
    },
    "Cayenne": {
      "exemplarCity": "Cayenne"
    },
    "Cayman": {
      "exemplarCity": "Kaimaninseln"
    },
    "Chicago": {
      "exemplarCity": "Chicago"
    },
    "Chihuahua": {
      "exemplarCity": "Chihuahua"
    },
    "Coral_Harbour": {
      "exemplarCity": "Atikokan"
    },
    "Cordoba": {
      "exemplarCity": "Córdoba"
    },
    "Costa_Rica": {
      "exemplarCity": "Costa Rica"
    },
    "Creston": {
      "exemplarCity": "Creston"
    },
    "Cuiaba": {
      "exemplarCity": "Cuiaba"
    },
    "Curacao": {
      "exemplarCity": "Curaçao"
    },
    "Danmarkshavn": {
      "exemplarCity": "Danmarkshavn"
    },
    "Dawson": {
      "exemplarCity": "Dawson"
    },
    "Dawson_Creek": {
      "exemplarCity": "Dawson Creek"
    },
    "Denver": {
      "exemplarCity": "Denver"
    },
    "Detroit": {
      "exemplarCity": "Detroit"
    },
    "Dominica": {
      "exemplarCity": "Dominica"
    },
    "Edmonton": {
      "exemplarCity": "Edmonton"
    },
    "Eirunepe": {
      "exemplarCity": "Eirunepe"
    },
    "El_Salvador": {
      "exemplarCity": "El Salvador"
    }
  }
```

```
    },
    "Fort_Nelson": {
      "exemplarCity": "Fort Nelson"
    },
    "Fortaleza": {
      "exemplarCity": "Fortaleza"
    },
    "Glace_Bay": {
      "exemplarCity": "Glace Bay"
    },
    "Godthab": {
      "exemplarCity": "Nuuk"
    },
    "Goose_Bay": {
      "exemplarCity": "Goose Bay"
    },
    "Grand_Turk": {
      "exemplarCity": "Grand Turk"
    },
    "Grenada": {
      "exemplarCity": "Grenada"
    },
    "Guadeloupe": {
      "exemplarCity": "Guadeloupe"
    },
    "Guatemala": {
      "exemplarCity": "Guatemala"
    },
    "Guayaquil": {
      "exemplarCity": "Guayaquil"
    },
    "Guyana": {
      "exemplarCity": "Guyana"
    },
    "Halifax": {
      "exemplarCity": "Halifax"
    },
    "Havana": {
      "exemplarCity": "Havanna"
    },
    "Hermosillo": {
      "exemplarCity": "Hermosillo"
    },
    "Indiana": {
      "Vincennes": {
        "exemplarCity": "Vincennes, Indiana"
      },
      "Petersburg": {
        "exemplarCity": "Petersburg, Indiana"
      },
      "Tell_City": {
        "exemplarCity": "Tell City, Indiana"
      },
      "Knox": {
        "exemplarCity": "Knox, Indiana"
      },
      "Winamac": {
```

```
        "exemplarCity": "Winamac, Indiana"
      },
      "Marengo": {
        "exemplarCity": "Marengo, Indiana"
      },
      "Vevay": {
        "exemplarCity": "Vevay, Indiana"
      }
    },
    "Indianapolis": {
      "exemplarCity": "Indianapolis"
    },
    "Inuvik": {
      "exemplarCity": "Inuvik"
    },
    "Iqaluit": {
      "exemplarCity": "Iqaluit"
    },
    "Jamaica": {
      "exemplarCity": "Jamaika"
    },
    "Jujuy": {
      "exemplarCity": "Jujuy"
    },
    "Juneau": {
      "exemplarCity": "Juneau"
    },
    "Kentucky": {
      "Monticello": {
        "exemplarCity": "Monticello, Kentucky"
      }
    },
    "Kralendijk": {
      "exemplarCity": "Kralendijk"
    },
    "La_Paz": {
      "exemplarCity": "La Paz"
    },
    "Lima": {
      "exemplarCity": "Lima"
    },
    "Los_Angeles": {
      "exemplarCity": "Los Angeles"
    },
    "Louisville": {
      "exemplarCity": "Louisville"
    },
    "Lower_Princes": {
      "exemplarCity": "Lower Prince's Quarter"
    },
    "Maceio": {
      "exemplarCity": "Maceio"
    },
    "Managua": {
      "exemplarCity": "Managua"
    },
    "Manaus": {
```

```
    "exemplarCity": "Manaus"
  },
  "Marigot": {
    "exemplarCity": "Marigot"
  },
  "Martinique": {
    "exemplarCity": "Martinique"
  },
  "Matamoros": {
    "exemplarCity": "Matamoros"
  },
  "Mazatlan": {
    "exemplarCity": "Mazatlan"
  },
  "Mendoza": {
    "exemplarCity": "Mendoza"
  },
  "Menominee": {
    "exemplarCity": "Menominee"
  },
  "Merida": {
    "exemplarCity": "Merida"
  },
  "Metlakatla": {
    "exemplarCity": "Metlakatla"
  },
  "Mexico_City": {
    "exemplarCity": "Mexiko-Stadt"
  },
  "Miquelon": {
    "exemplarCity": "Miquelon"
  },
  "Moncton": {
    "exemplarCity": "Moncton"
  },
  "Monterrey": {
    "exemplarCity": "Monterrey"
  },
  "Montevideo": {
    "exemplarCity": "Montevideo"
  },
  "Montserrat": {
    "exemplarCity": "Montserrat"
  },
  "Nassau": {
    "exemplarCity": "Nassau"
  },
  "New_York": {
    "exemplarCity": "New York"
  },
  "Nipigon": {
    "exemplarCity": "Nipigon"
  },
  "Nome": {
    "exemplarCity": "Nome"
  },
  "Noronha": {
```

```
    "exemplarCity": "Noronha"
  },
  "North_Dakota": {
    "Beulah": {
      "exemplarCity": "Beulah, North Dakota"
    },
    "New_Salem": {
      "exemplarCity": "New Salem, North Dakota"
    },
    "Center": {
      "exemplarCity": "Center, North Dakota"
    }
  },
  "Ojinaga": {
    "exemplarCity": "Ojinaga"
  },
  "Panama": {
    "exemplarCity": "Panama"
  },
  "Pangnirtung": {
    "exemplarCity": "Pangnirtung"
  },
  "Paramaribo": {
    "exemplarCity": "Paramaribo"
  },
  "Phoenix": {
    "exemplarCity": "Phoenix"
  },
  "Port-au-Prince": {
    "exemplarCity": "Port-au-Prince"
  },
  "Port_of_Spain": {
    "exemplarCity": "Port of Spain"
  },
  "Porto_Velho": {
    "exemplarCity": "Porto Velho"
  },
  "Puerto_Rico": {
    "exemplarCity": "Puerto Rico"
  },
  "Rainy_River": {
    "exemplarCity": "Rainy River"
  },
  "Rankin_Inlet": {
    "exemplarCity": "Rankin Inlet"
  },
  "Recife": {
    "exemplarCity": "Recife"
  },
  "Regina": {
    "exemplarCity": "Regina"
  },
  "Resolute": {
    "exemplarCity": "Resolute"
  },
  "Rio_Branco": {
    "exemplarCity": "Rio Branco"
  }
```

```
    },
    "Santa_Isabel": {
      "exemplarCity": "Santa Isabel"
    },
    "Santarem": {
      "exemplarCity": "Santarem"
    },
    "Santiago": {
      "exemplarCity": "Santiago"
    },
    "Santo_Domingo": {
      "exemplarCity": "Santo Domingo"
    },
    "Sao_Paulo": {
      "exemplarCity": "São Paulo"
    },
    "Scoresbysund": {
      "exemplarCity": "Ittoqqortoormiit"
    },
    "Sitka": {
      "exemplarCity": "Sitka"
    },
    "St_Barthelemy": {
      "exemplarCity": "Saint-Barthélemy"
    },
    "St_Johns": {
      "exemplarCity": "St. John's"
    },
    "St_Kitts": {
      "exemplarCity": "St. Kitts"
    },
    "St_Lucia": {
      "exemplarCity": "St. Lucia"
    },
    "St_Thomas": {
      "exemplarCity": "St. Thomas"
    },
    "St_Vincent": {
      "exemplarCity": "St. Vincent"
    },
    "Swift_Current": {
      "exemplarCity": "Swift Current"
    },
    "Tegucigalpa": {
      "exemplarCity": "Tegucigalpa"
    },
    "Thule": {
      "exemplarCity": "Thule"
    },
    "Thunder_Bay": {
      "exemplarCity": "Thunder Bay"
    },
    "Tijuana": {
      "exemplarCity": "Tijuana"
    },
    "Toronto": {
      "exemplarCity": "Toronto"
    }
  }
```



```

    },
    "Tortola": {
      "exemplarCity": "Tortola"
    },
    "Vancouver": {
      "exemplarCity": "Vancouver"
    },
    "Whitehorse": {
      "exemplarCity": "Whitehorse"
    },
    "Winnipeg": {
      "exemplarCity": "Winnipeg"
    },
    "Yakutat": {
      "exemplarCity": "Yakutat"
    },
    "Yellowknife": {
      "exemplarCity": "Yellowknife"
    }
  },
  "Atlantic": {
    "Azores": {
      "exemplarCity": "Azoren"
    },
    "Bermuda": {
      "exemplarCity": "Bermudas"
    },
    "Canary": {
      "exemplarCity": "Kanaren"
    },
    "Cape_Verde": {
      "exemplarCity": "Cabo Verde"
    },
    "Faeroe": {
      "exemplarCity": "Färöer"
    },
    "Madeira": {
      "exemplarCity": "Madeira"
    },
    "Reykjavik": {
      "exemplarCity": "Reykjavík"
    },
    "South_Georgia": {
      "exemplarCity": "Südgeorgien"
    },
    "St_Helena": {
      "exemplarCity": "St. Helena"
    },
    "Stanley": {
      "exemplarCity": "Stanley"
    }
  },
  "Europe": {
    "Amsterdam": {
      "exemplarCity": "Amsterdam"
    },
    "Andorra": {

```

```
    "exemplarCity": "Andorra"
  },
  "Astrakhan": {
    "exemplarCity": "Astrachan"
  },
  "Athens": {
    "exemplarCity": "Athen"
  },
  "Belgrade": {
    "exemplarCity": "Belgrad"
  },
  "Berlin": {
    "exemplarCity": "Berlin"
  },
  "Bratislava": {
    "exemplarCity": "Bratislava"
  },
  "Brussels": {
    "exemplarCity": "Brüssel"
  },
  "Bucharest": {
    "exemplarCity": "Bukarest"
  },
  "Budapest": {
    "exemplarCity": "Budapest"
  },
  "Busingen": {
    "exemplarCity": "Büsingen"
  },
  "Chisinau": {
    "exemplarCity": "Kischinau"
  },
  "Copenhagen": {
    "exemplarCity": "Kopenhagen"
  },
  "Dublin": {
    "long": {
      "daylight": "Irische Sommerzeit"
    },
    "exemplarCity": "Dublin"
  },
  "Gibraltar": {
    "exemplarCity": "Gibraltar"
  },
  "Guernsey": {
    "exemplarCity": "Guernsey"
  },
  "Helsinki": {
    "exemplarCity": "Helsinki"
  },
  "Isle_of_Man": {
    "exemplarCity": "Isle of Man"
  },
  "Istanbul": {
    "exemplarCity": "Istanbul"
  },
  "Jersey": {
```

```
        "exemplarCity": "Jersey"
    },
    "Kaliningrad": {
        "exemplarCity": "Kaliningrad"
    },
    "Kiev": {
        "exemplarCity": "Kiew"
    },
    "Kirov": {
        "exemplarCity": "Kirow"
    },
    "Lisbon": {
        "exemplarCity": "Lissabon"
    },
    "Ljubljana": {
        "exemplarCity": "Ljubljana"
    },
    "London": {
        "long": {
            "daylight": "Britische Sommerzeit"
        },
        "exemplarCity": "London"
    },
    "Luxembourg": {
        "exemplarCity": "Luxemburg"
    },
    "Madrid": {
        "exemplarCity": "Madrid"
    },
    "Malta": {
        "exemplarCity": "Malta"
    },
    "Mariehamn": {
        "exemplarCity": "Mariehamn"
    },
    "Minsk": {
        "exemplarCity": "Minsk"
    },
    "Monaco": {
        "exemplarCity": "Monaco"
    },
    "Moscow": {
        "exemplarCity": "Moskau"
    },
    "Oslo": {
        "exemplarCity": "Oslo"
    },
    "Paris": {
        "exemplarCity": "Paris"
    },
    "Podgorica": {
        "exemplarCity": "Podgorica"
    },
    "Prague": {
        "exemplarCity": "Prag"
    },
    "Riga": {
```

```
        "exemplarCity": "Riga"
      },
      "Rome": {
        "exemplarCity": "Rom"
      },
      "Samara": {
        "exemplarCity": "Samara"
      },
      "San_Marino": {
        "exemplarCity": "San Marino"
      },
      "Sarajevo": {
        "exemplarCity": "Sarajevo"
      },
      "Simferopol": {
        "exemplarCity": "Simferopol"
      },
      "Skopje": {
        "exemplarCity": "Skopje"
      },
      "Sofia": {
        "exemplarCity": "Sofia"
      },
      "Stockholm": {
        "exemplarCity": "Stockholm"
      },
      "Tallinn": {
        "exemplarCity": "Tallinn"
      },
      "Tirane": {
        "exemplarCity": "Tirana"
      },
      "Ulyanovsk": {
        "exemplarCity": "Uljanowsk"
      },
      "Uzhgorod": {
        "exemplarCity": "Uschgorod"
      },
      "Vaduz": {
        "exemplarCity": "Vaduz"
      },
      "Vatican": {
        "exemplarCity": "Vatikan"
      },
      "Vienna": {
        "exemplarCity": "Wien"
      },
      "Vilnius": {
        "exemplarCity": "Vilnius"
      },
      "Volgograd": {
        "exemplarCity": "Wolgograd"
      },
      "Warsaw": {
        "exemplarCity": "Warschau"
      },
      "Zagreb": {
```

```
        "exemplarCity": "Zagreb"
      },
      "Zaporozhye": {
        "exemplarCity": "Saporischja"
      },
      "Zurich": {
        "exemplarCity": "Zürich"
      }
    },
    "Africa": {
      "Abidjan": {
        "exemplarCity": "Abidjan"
      },
      "Accra": {
        "exemplarCity": "Accra"
      },
      "Addis_Ababa": {
        "exemplarCity": "Addis Abeba"
      },
      "Algiers": {
        "exemplarCity": "Algier"
      },
      "Asmera": {
        "exemplarCity": "Asmara"
      },
      "Bamako": {
        "exemplarCity": "Bamako"
      },
      "Bangui": {
        "exemplarCity": "Bangui"
      },
      "Banjul": {
        "exemplarCity": "Banjul"
      },
      "Bissau": {
        "exemplarCity": "Bissau"
      },
      "Blantyre": {
        "exemplarCity": "Blantyre"
      },
      "Brazzaville": {
        "exemplarCity": "Brazzaville"
      },
      "Bujumbura": {
        "exemplarCity": "Bujumbura"
      },
      "Cairo": {
        "exemplarCity": "Kairo"
      },
      "Casablanca": {
        "exemplarCity": "Casablanca"
      },
      "Ceuta": {
        "exemplarCity": "Ceuta"
      },
      "Conakry": {
        "exemplarCity": "Conakry"
      }
    }
  }
}
```

```
    },  
    "Dakar": {  
      "exemplarCity": "Dakar"  
    },  
    "Dar_es_Salaam": {  
      "exemplarCity": "Daressalam"  
    },  
    "Djibouti": {  
      "exemplarCity": "Dschibuti"  
    },  
    "Douala": {  
      "exemplarCity": "Douala"  
    },  
    "El_Aaiun": {  
      "exemplarCity": "El Aaiún"  
    },  
    "Freetown": {  
      "exemplarCity": "Freetown"  
    },  
    "Gaborone": {  
      "exemplarCity": "Gaborone"  
    },  
    "Harare": {  
      "exemplarCity": "Harare"  
    },  
    "Johannesburg": {  
      "exemplarCity": "Johannesburg"  
    },  
    "Juba": {  
      "exemplarCity": "Juba"  
    },  
    "Kampala": {  
      "exemplarCity": "Kampala"  
    },  
    "Khartoum": {  
      "exemplarCity": "Khartum"  
    },  
    "Kigali": {  
      "exemplarCity": "Kigali"  
    },  
    "Kinshasa": {  
      "exemplarCity": "Kinshasa"  
    },  
    "Lagos": {  
      "exemplarCity": "Lagos"  
    },  
    "Libreville": {  
      "exemplarCity": "Libreville"  
    },  
    "Lome": {  
      "exemplarCity": "Lomé"  
    },  
    "Luanda": {  
      "exemplarCity": "Luanda"  
    },  
    "Lubumbashi": {  
      "exemplarCity": "Lubumbashi"
```

```
    },  
    "Lusaka": {  
      "exemplarCity": "Lusaka"  
    },  
    "Malabo": {  
      "exemplarCity": "Malabo"  
    },  
    "Maputo": {  
      "exemplarCity": "Maputo"  
    },  
    "Maseru": {  
      "exemplarCity": "Maseru"  
    },  
    "Mbabane": {  
      "exemplarCity": "Mbabane"  
    },  
    "Mogadishu": {  
      "exemplarCity": "Mogadischu"  
    },  
    "Monrovia": {  
      "exemplarCity": "Monrovia"  
    },  
    "Nairobi": {  
      "exemplarCity": "Nairobi"  
    },  
    "Ndjamena": {  
      "exemplarCity": "N'Djamena"  
    },  
    "Niamey": {  
      "exemplarCity": "Niamey"  
    },  
    "Nouakchott": {  
      "exemplarCity": "Nouakchott"  
    },  
    "Ouagadougou": {  
      "exemplarCity": "Ouagadougou"  
    },  
    "Porto-Novo": {  
      "exemplarCity": "Porto Novo"  
    },  
    "Sao_Tome": {  
      "exemplarCity": "São Tomé"  
    },  
    "Tripoli": {  
      "exemplarCity": "Tripolis"  
    },  
    "Tunis": {  
      "exemplarCity": "Tunis"  
    },  
    "Windhoek": {  
      "exemplarCity": "Windhoek"  
    }  
  },  
  "Asia": {  
    "Aden": {  
      "exemplarCity": "Aden"  
    },  
  },  
}
```

```
"Almaty": {
  "exemplarCity": "Almaty"
},
"Amman": {
  "exemplarCity": "Amman"
},
"Anadyr": {
  "exemplarCity": "Anadyr"
},
"Aqtau": {
  "exemplarCity": "Aqtau"
},
"Aqtobe": {
  "exemplarCity": "Aktobe"
},
"Ashgabat": {
  "exemplarCity": "Aşgabat"
},
"Baghdad": {
  "exemplarCity": "Bagdad"
},
"Bahrain": {
  "exemplarCity": "Bahrain"
},
"Baku": {
  "exemplarCity": "Baku"
},
"Bangkok": {
  "exemplarCity": "Bangkok"
},
"Barnaul": {
  "exemplarCity": "Barnaul"
},
"Beirut": {
  "exemplarCity": "Beirut"
},
"Bishkek": {
  "exemplarCity": "Bischkek"
},
"Brunei": {
  "exemplarCity": "Brunei"
},
"Calcutta": {
  "exemplarCity": "Kalkutta"
},
"Chita": {
  "exemplarCity": "Tschita"
},
"Choibalsan": {
  "exemplarCity": "Tschoibalsan"
},
"Colombo": {
  "exemplarCity": "Colombo"
},
"Damascus": {
  "exemplarCity": "Damaskus"
},
```



```
"Dhaka": {
  "exemplarCity": "Dhaka"
},
"Dili": {
  "exemplarCity": "Dili"
},
"Dubai": {
  "exemplarCity": "Dubai"
},
"Dushanbe": {
  "exemplarCity": "Duschanbe"
},
"Gaza": {
  "exemplarCity": "Gaza"
},
"Hebron": {
  "exemplarCity": "Hebron"
},
"Hong_Kong": {
  "exemplarCity": "Hongkong"
},
"Hovd": {
  "exemplarCity": "Chowd"
},
"Irkutsk": {
  "exemplarCity": "Irkutsk"
},
"Jakarta": {
  "exemplarCity": "Jakarta"
},
"Jayapura": {
  "exemplarCity": "Jayapura"
},
"Jerusalem": {
  "exemplarCity": "Jerusalem"
},
"Kabul": {
  "exemplarCity": "Kabul"
},
"Kamchatka": {
  "exemplarCity": "Kamtschatka"
},
"Karachi": {
  "exemplarCity": "Karatschi"
},
"Katmandu": {
  "exemplarCity": "Kathmandu"
},
"Khandyga": {
  "exemplarCity": "Chandyga"
},
"Krasnoyarsk": {
  "exemplarCity": "Krasnojarsk"
},
"Kuala_Lumpur": {
  "exemplarCity": "Kuala Lumpur"
},
}
```

```
"Kuching": {
  "exemplarCity": "Kuching"
},
"Kuwait": {
  "exemplarCity": "Kuwait"
},
"Macau": {
  "exemplarCity": "Macao"
},
"Magadan": {
  "exemplarCity": "Magadan"
},
"Makassar": {
  "exemplarCity": "Makassar"
},
"Manila": {
  "exemplarCity": "Manila"
},
"Muscat": {
  "exemplarCity": "Maskat"
},
"Nicosia": {
  "exemplarCity": "Nikosia"
},
"Novokuznetsk": {
  "exemplarCity": "Nowokuznetsk"
},
"Novosibirsk": {
  "exemplarCity": "Nowosibirsk"
},
"Omsk": {
  "exemplarCity": "Omsk"
},
"Oral": {
  "exemplarCity": "Oral"
},
"Phnom_Penh": {
  "exemplarCity": "Phnom Penh"
},
"Pontianak": {
  "exemplarCity": "Pontianak"
},
"Pyongyang": {
  "exemplarCity": "Pjöngjang"
},
"Qatar": {
  "exemplarCity": "Katar"
},
"Qyzylorda": {
  "exemplarCity": "Qysylorda"
},
"Rangoon": {
  "exemplarCity": "Rangun"
},
"Riyadh": {
  "exemplarCity": "Riad"
},
}
```

```
"Saigon": {
  "exemplarCity": "Ho-Chi-Minh-Stadt"
},
"Sakhalin": {
  "exemplarCity": "Sachalin"
},
"Samarkand": {
  "exemplarCity": "Samarkand"
},
"Seoul": {
  "exemplarCity": "Seoul"
},
"Shanghai": {
  "exemplarCity": "Shanghai"
},
"Singapore": {
  "exemplarCity": "Singapur"
},
"Srednekolymsk": {
  "exemplarCity": "Srednekolymsk"
},
"Taipei": {
  "exemplarCity": "Taipeh"
},
"Tashkent": {
  "exemplarCity": "Taschkent"
},
"Tbilisi": {
  "exemplarCity": "Tiflis"
},
"Tehran": {
  "exemplarCity": "Teheran"
},
"Thimphu": {
  "exemplarCity": "Thimphu"
},
"Tokyo": {
  "exemplarCity": "Tokio"
},
"Tomsk": {
  "exemplarCity": "Tomsk"
},
"Ulaanbaatar": {
  "exemplarCity": "Ulaanbaatar"
},
"Urumqi": {
  "exemplarCity": "Ürümqi"
},
"Ust-Nera": {
  "exemplarCity": "Ust-Nera"
},
"Vientiane": {
  "exemplarCity": "Vientiane"
},
"Vladivostok": {
  "exemplarCity": "Wladiwostok"
},
```

```
"Yakutsk": {
  "exemplarCity": "Jakutsk"
},
"Yekaterinburg": {
  "exemplarCity": "Jekaterinburg"
},
"Yerevan": {
  "exemplarCity": "Eriwan"
}
},
"Indian": {
  "Antananarivo": {
    "exemplarCity": "Antananarivo"
  },
  "Chagos": {
    "exemplarCity": "Chagos"
  },
  "Christmas": {
    "exemplarCity": "Weihnachtsinsel"
  },
  "Cocos": {
    "exemplarCity": "Cocos"
  },
  "Comoro": {
    "exemplarCity": "Komoren"
  },
  "Kerguelen": {
    "exemplarCity": "Kerguelen"
  },
  "Mahe": {
    "exemplarCity": "Mahe"
  },
  "Maldives": {
    "exemplarCity": "Malediven"
  },
  "Mauritius": {
    "exemplarCity": "Mauritius"
  },
  "Mayotte": {
    "exemplarCity": "Mayotte"
  },
  "Reunion": {
    "exemplarCity": "Réunion"
  }
},
"Australia": {
  "Adelaide": {
    "exemplarCity": "Adelaide"
  },
  "Brisbane": {
    "exemplarCity": "Brisbane"
  },
  "Broken_Hill": {
    "exemplarCity": "Broken Hill"
  },
  "Currie": {
    "exemplarCity": "Currie"
  }
}
```

```
    },
    "Darwin": {
      "exemplarCity": "Darwin"
    },
    "Eucla": {
      "exemplarCity": "Eucla"
    },
    "Hobart": {
      "exemplarCity": "Hobart"
    },
    "Lindeman": {
      "exemplarCity": "Lindeman"
    },
    "Lord_Howe": {
      "exemplarCity": "Lord Howe"
    },
    "Melbourne": {
      "exemplarCity": "Melbourne"
    },
    "Perth": {
      "exemplarCity": "Perth"
    },
    "Sydney": {
      "exemplarCity": "Sydney"
    }
  },
  "Pacific": {
    "Apia": {
      "exemplarCity": "Apia"
    },
    "Auckland": {
      "exemplarCity": "Auckland"
    },
    "Bougainville": {
      "exemplarCity": "Bougainville"
    },
    "Chatham": {
      "exemplarCity": "Chatham"
    },
    "Easter": {
      "exemplarCity": "Osterinsel"
    },
    "Efate": {
      "exemplarCity": "Efate"
    },
    "Enderbury": {
      "exemplarCity": "Enderbury"
    },
    "Fakaofu": {
      "exemplarCity": "Fakaofu"
    },
    "Fiji": {
      "exemplarCity": "Fidschi"
    },
    "Funafuti": {
      "exemplarCity": "Funafuti"
    }
  },
}
```

```
"Galapagos": {
  "exemplarCity": "Galapagos"
},
"Gambier": {
  "exemplarCity": "Gambier"
},
"Guadalcanal": {
  "exemplarCity": "Guadalcanal"
},
"Guam": {
  "exemplarCity": "Guam"
},
"Honolulu": {
  "exemplarCity": "Honolulu"
},
"Johnston": {
  "exemplarCity": "Johnston"
},
"Kiritimati": {
  "exemplarCity": "Kiritimati"
},
"Kosrae": {
  "exemplarCity": "Kosrae"
},
"Kwajalein": {
  "exemplarCity": "Kwajalein"
},
"Majuro": {
  "exemplarCity": "Majuro"
},
"Marquesas": {
  "exemplarCity": "Marquesas"
},
"Midway": {
  "exemplarCity": "Midway"
},
"Nauru": {
  "exemplarCity": "Nauru"
},
"Niue": {
  "exemplarCity": "Niue"
},
"Norfolk": {
  "exemplarCity": "Norfolk"
},
"Noumea": {
  "exemplarCity": "Noumea"
},
"Pago_Pago": {
  "exemplarCity": "Pago Pago"
},
"Palau": {
  "exemplarCity": "Palau"
},
"Pitcairn": {
  "exemplarCity": "Pitcairn"
},
}
```

```
"Ponape": {
  "exemplarCity": "Pohnpei"
},
"Port_Moresby": {
  "exemplarCity": "Port Moresby"
},
"Rarotonga": {
  "exemplarCity": "Rarotonga"
},
"Saipan": {
  "exemplarCity": "Saipan"
},
"Tahiti": {
  "exemplarCity": "Tahiti"
},
"Tarawa": {
  "exemplarCity": "Tarawa"
},
"Tongatapu": {
  "exemplarCity": "Tongatapu"
},
"Truk": {
  "exemplarCity": "Chuuk"
},
"Wake": {
  "exemplarCity": "Wake"
},
"Wallis": {
  "exemplarCity": "Wallis"
}
},
"Arctic": {
  "Longyearbyen": {
    "exemplarCity": "Longyearbyen"
  }
},
"Antarctica": {
  "Casey": {
    "exemplarCity": "Casey"
  },
  "Davis": {
    "exemplarCity": "Davis"
  },
  "DumontDUrville": {
    "exemplarCity": "Dumont d'Urville"
  },
  "Macquarie": {
    "exemplarCity": "Macquarie"
  },
  "Mawson": {
    "exemplarCity": "Mawson"
  },
  "McMurdo": {
    "exemplarCity": "McMurdo"
  },
  "Palmer": {
    "exemplarCity": "Palmer"
  }
}
```

```
    },  
    "Rothera": {  
      "exemplarCity": "Rothera"  
    },  
    "Syowa": {  
      "exemplarCity": "Syowa"  
    },  
    "Troll": {  
      "exemplarCity": "Troll"  
    },  
    "Vostok": {  
      "exemplarCity": "Wostok"  
    }  
  },  
  "Etc": {  
    "GMT": {  
      "exemplarCity": "GMT"  
    },  
    "GMT1": {  
      "exemplarCity": "GMT+1"  
    },  
    "GMT10": {  
      "exemplarCity": "GMT+10"  
    },  
    "GMT11": {  
      "exemplarCity": "GMT+11"  
    },  
    "GMT12": {  
      "exemplarCity": "GMT+12"  
    },  
    "GMT2": {  
      "exemplarCity": "GMT+2"  
    },  
    "GMT3": {  
      "exemplarCity": "GMT+3"  
    },  
    "GMT4": {  
      "exemplarCity": "GMT+4"  
    },  
    "GMT5": {  
      "exemplarCity": "GMT+5"  
    },  
    "GMT6": {  
      "exemplarCity": "GMT+6"  
    },  
    "GMT7": {  
      "exemplarCity": "GMT+7"  
    },  
    "GMT8": {  
      "exemplarCity": "GMT+8"  
    },  
    "GMT9": {  
      "exemplarCity": "GMT+9"  
    },  
    "GMT-1": {  
      "exemplarCity": "GMT-1"  
    },  
  },  
}
```



```

    "GMT-10": {
      "exemplarCity": "GMT-10"
    },
    "GMT-11": {
      "exemplarCity": "GMT-11"
    },
    "GMT-12": {
      "exemplarCity": "GMT-12"
    },
    "GMT-13": {
      "exemplarCity": "GMT-13"
    },
    "GMT-14": {
      "exemplarCity": "GMT-14"
    },
    "GMT-2": {
      "exemplarCity": "GMT-2"
    },
    "GMT-3": {
      "exemplarCity": "GMT-3"
    },
    "GMT-4": {
      "exemplarCity": "GMT-4"
    },
    "GMT-5": {
      "exemplarCity": "GMT-5"
    },
    "GMT-6": {
      "exemplarCity": "GMT-6"
    },
    "GMT-7": {
      "exemplarCity": "GMT-7"
    },
    "GMT-8": {
      "exemplarCity": "GMT-8"
    },
    "GMT-9": {
      "exemplarCity": "GMT-9"
    },
    "Unknown": {
      "exemplarCity": "Unbekannt"
    }
  },
  "metazone": {
    "Acre": {
      "long": {
        "generic": "Acre-Zeit",
        "standard": "Acre-Normalzeit",
        "daylight": "Acre-Sommerzeit"
      }
    },
    "Afghanistan": {
      "long": {
        "standard": "Afghanistan-Zeit"
      }
    }
  },

```

```
"Africa_Central": {
  "long": {
    "standard": "Zentralafrikanische Zeit"
  }
},
"Africa_Eastern": {
  "long": {
    "standard": "Ostafrikanische Zeit"
  }
},
"Africa_Southern": {
  "long": {
    "standard": "Südafrikanische Zeit"
  }
},
"Africa_Western": {
  "long": {
    "generic": "Westafrikanische Zeit",
    "standard": "Westafrikanische Normalzeit",
    "daylight": "Westafrikanische Sommerzeit"
  }
},
"Alaska": {
  "long": {
    "generic": "Alaska-Zeit",
    "standard": "Alaska-Normalzeit",
    "daylight": "Alaska-Sommerzeit"
  }
},
"Almaty": {
  "long": {
    "generic": "Almaty-Zeit",
    "standard": "Almaty-Normalzeit",
    "daylight": "Almaty-Sommerzeit"
  }
},
"Amazon": {
  "long": {
    "generic": "Amazonas-Zeit",
    "standard": "Amazonas-Normalzeit",
    "daylight": "Amazonas-Sommerzeit"
  }
},
"America_Central": {
  "long": {
    "generic": "Nordamerikanische Inlandzeit",
    "standard": "Nordamerikanische Inland-Normalzeit",
    "daylight": "Nordamerikanische Inland-Sommerzeit"
  }
},
"America_Eastern": {
  "long": {
    "generic": "Nordamerikanische Ostküstenzeit",
    "standard": "Nordamerikanische Ostküsten-Normalzeit",
    "daylight": "Nordamerikanische Ostküsten-Sommerzeit"
  }
},
}
```

```
"America_Mountain": {
  "long": {
    "generic": "Rocky-Mountain-Zeit",
    "standard": "Rocky Mountain-Normalzeit",
    "daylight": "Rocky-Mountain-Sommerzeit"
  }
},
"America_Pacific": {
  "long": {
    "generic": "Nordamerikanische Westküstenzeit",
    "standard": "Nordamerikanische Westküsten-Normalzeit",
    "daylight": "Nordamerikanische Westküsten-Sommerzeit"
  }
},
"Anadyr": {
  "long": {
    "generic": "Anadyr Zeit",
    "standard": "Anadyr Normalzeit",
    "daylight": "Anadyr Sommerzeit"
  }
},
"Apia": {
  "long": {
    "generic": "Apia-Zeit",
    "standard": "Apia-Normalzeit",
    "daylight": "Apia-Sommerzeit"
  }
},
"Aqtau": {
  "long": {
    "generic": "Aqtau-Zeit",
    "standard": "Aqtau-Normalzeit",
    "daylight": "Aqtau-Sommerzeit"
  }
},
"Aqtobe": {
  "long": {
    "generic": "Aqtöbe-Zeit",
    "standard": "Aqtöbe-Normalzeit",
    "daylight": "Aqtöbe-Sommerzeit"
  }
},
"Arabian": {
  "long": {
    "generic": "Arabische Zeit",
    "standard": "Arabische Normalzeit",
    "daylight": "Arabische Sommerzeit"
  }
},
"Argentina": {
  "long": {
    "generic": "Argentinische Zeit",
    "standard": "Argentinische Normalzeit",
    "daylight": "Argentinische Sommerzeit"
  }
},
"Argentina_Western": {
```

```

        "long": {
            "generic": "Westargentinische Zeit",
            "standard": "Westargentinische Normalzeit",
            "daylight": "Westargentinische Sommerzeit"
        }
    },
    "Armenia": {
        "long": {
            "generic": "Armenische Zeit",
            "standard": "Armenische Normalzeit",
            "daylight": "Armenische Sommerzeit"
        }
    },
    "Atlantic": {
        "long": {
            "generic": "Atlantik-Zeit",
            "standard": "Atlantik-Normalzeit",
            "daylight": "Atlantik-Sommerzeit"
        }
    },
    "Australia_Central": {
        "long": {
            "generic": "Zentralaustralische Zeit",
            "standard": "Zentralaustralische Normalzeit",
            "daylight": "Zentralaustralische Sommerzeit"
        }
    },
    "Australia_CentralWestern": {
        "long": {
            "generic": "Zentral-/Westaustralische Zeit",
            "standard": "Zentral-/Westaustralische Normalzeit",
            "daylight": "Zentral-/Westaustralische Sommerzeit"
        }
    },
    "Australia_Eastern": {
        "long": {
            "generic": "Ostaustralische Zeit",
            "standard": "Ostaustralische Normalzeit",
            "daylight": "Ostaustralische Sommerzeit"
        }
    },
    "Australia_Western": {
        "long": {
            "generic": "Westaustralische Zeit",
            "standard": "Westaustralische Normalzeit",
            "daylight": "Westaustralische Sommerzeit"
        }
    },
    "Azerbaijan": {
        "long": {
            "generic": "Aserbaidtschanische Zeit",
            "standard": "Aserbeidschanische Normalzeit",
            "daylight": "Aserbaidtschanische Sommerzeit"
        }
    },
    "Azores": {
        "long": {

```

```

        "generic": "Azoren-Zeit",
        "standard": "Azoren-Normalzeit",
        "daylight": "Azoren-Sommerzeit"
    },
    },
    "Bangladesh": {
        "long": {
            "generic": "Bangladesch-Zeit",
            "standard": "Bangladesch-Normalzeit",
            "daylight": "Bangladesch-Sommerzeit"
        }
    },
    },
    "Bhutan": {
        "long": {
            "standard": "Bhutan-Zeit"
        }
    },
    },
    "Bolivia": {
        "long": {
            "standard": "Bolivianische Zeit"
        }
    },
    },
    "Brasilia": {
        "long": {
            "generic": "Brasília-Zeit",
            "standard": "Brasília-Normalzeit",
            "daylight": "Brasília-Sommerzeit"
        }
    },
    },
    "Brunei": {
        "long": {
            "standard": "Brunei-Zeit"
        }
    },
    },
    "Cape_Verde": {
        "long": {
            "generic": "Cabo-Verde-Zeit",
            "standard": "Cabo-Verde-Normalzeit",
            "daylight": "Cabo-Verde-Sommerzeit"
        }
    },
    },
    "Casey": {
        "long": {
            "standard": "Casey-Zeit"
        }
    },
    },
    "Chamorro": {
        "long": {
            "standard": "Chamorro-Zeit"
        }
    },
    },
    "Chatham": {
        "long": {
            "generic": "Chatham-Zeit",
            "standard": "Chatham-Normalzeit",
            "daylight": "Chatham-Sommerzeit"
        }
    }
}

```

```
    },  
    "Chile": {  
      "long": {  
        "generic": "Chilenische Zeit",  
        "standard": "Chilenische Normalzeit",  
        "daylight": "Chilenische Sommerzeit"  
      }  
    },  
    "China": {  
      "long": {  
        "generic": "Chinesische Zeit",  
        "standard": "Chinesische Normalzeit",  
        "daylight": "Chinesische Sommerzeit"  
      }  
    },  
    "Choibalsan": {  
      "long": {  
        "generic": "Tschoibalsan-Zeit",  
        "standard": "Tschoibalsan-Normalzeit",  
        "daylight": "Tschoibalsan-Sommerzeit"  
      }  
    },  
    "Christmas": {  
      "long": {  
        "standard": "Weihnachtsinsel-Zeit"  
      }  
    },  
    "Cocos": {  
      "long": {  
        "standard": "Kokosinseln-Zeit"  
      }  
    },  
    "Colombia": {  
      "long": {  
        "generic": "Kolumbianische Zeit",  
        "standard": "Kolumbianische Normalzeit",  
        "daylight": "Kolumbianische Sommerzeit"  
      }  
    },  
    "Cook": {  
      "long": {  
        "generic": "Cookinseln-Zeit",  
        "standard": "Cookinseln-Normalzeit",  
        "daylight": "Cookinseln-Sommerzeit"  
      }  
    },  
    "Cuba": {  
      "long": {  
        "generic": "Kubanische Zeit",  
        "standard": "Kubanische Normalzeit",  
        "daylight": "Kubanische Sommerzeit"  
      }  
    },  
    "Davis": {  
      "long": {  
        "standard": "Davis-Zeit"  
      }  
    }  
  }  
}
```

```

    },
    "DumontDUrville": {
      "long": {
        "standard": "Dumont-d'Urville-Zeit"
      }
    },
    "East_Timor": {
      "long": {
        "standard": "Osttimor-Zeit"
      }
    },
    "Easter": {
      "long": {
        "generic": "Osterinsel-Zeit",
        "standard": "Osterinsel-Normalzeit",
        "daylight": "Osterinsel-Sommerzeit"
      }
    },
    "Ecuador": {
      "long": {
        "standard": "Ecuadorianische Zeit"
      }
    },
    "Europe_Central": {
      "long": {
        "generic": "Mittleuropäische Zeit",
        "standard": "Mittleuropäische Normalzeit",
        "daylight": "Mittleuropäische Sommerzeit"
      },
      "short": {
        "generic": "MEZ",
        "standard": "MEZ",
        "daylight": "MESZ"
      }
    },
    "Europe_Eastern": {
      "long": {
        "generic": "Osteuropäische Zeit",
        "standard": "Osteuropäische Normalzeit",
        "daylight": "Osteuropäische Sommerzeit"
      },
      "short": {
        "generic": "OEZ",
        "standard": "OEZ",
        "daylight": "OESZ"
      }
    },
    "Europe_Further_Eastern": {
      "long": {
        "standard": "Kaliningrader Zeit"
      }
    },
    "Europe_Western": {
      "long": {
        "generic": "Westeuropäische Zeit",
        "standard": "Westeuropäische Normalzeit",
        "daylight": "Westeuropäische Sommerzeit"
      }
    }
  }

```

```
    },
    "short": {
      "generic": "WEZ",
      "standard": "WEZ",
      "daylight": "WESZ"
    }
  },
  "Falkland": {
    "long": {
      "generic": "Falklandinseln-Zeit",
      "standard": "Falklandinseln-Normalzeit",
      "daylight": "Falklandinseln-Sommerzeit"
    }
  },
  "Fiji": {
    "long": {
      "generic": "Fidschi-Zeit",
      "standard": "Fidschi-Normalzeit",
      "daylight": "Fidschi-Sommerzeit"
    }
  },
  "French_Guiana": {
    "long": {
      "standard": "Französisch-Guayana-Zeit"
    }
  },
  "French_Southern": {
    "long": {
      "standard": "Französische Süd- und Antarktisgebiete-Zeit"
    }
  },
  "Galapagos": {
    "long": {
      "standard": "Galapagos-Zeit"
    }
  },
  "Gambier": {
    "long": {
      "standard": "Gambier-Zeit"
    }
  },
  "Georgia": {
    "long": {
      "generic": "Georgische Zeit",
      "standard": "Georgische Normalzeit",
      "daylight": "Georgische Sommerzeit"
    }
  },
  "Gilbert_Islands": {
    "long": {
      "standard": "Gilbert-Inseln-Zeit"
    }
  },
  "GMT": {
    "long": {
      "standard": "Mittlere Greenwich-Zeit"
    }
  }
}
```



```

    },
    "Greenland_Eastern": {
      "long": {
        "generic": "Ostgrönland-Zeit",
        "standard": "Ostgrönland-Normalzeit",
        "daylight": "Ostgrönland-Sommerzeit"
      }
    },
    "Greenland_Western": {
      "long": {
        "generic": "Westgrönland-Zeit",
        "standard": "Westgrönland-Normalzeit",
        "daylight": "Westgrönland-Sommerzeit"
      }
    },
    "Guam": {
      "long": {
        "standard": "Guam-Zeit"
      }
    },
    "Gulf": {
      "long": {
        "standard": "Golf-Zeit"
      }
    },
    "Guyana": {
      "long": {
        "standard": "Guyana-Zeit"
      }
    },
    "Hawaii_Aleutian": {
      "long": {
        "generic": "Hawaii-Aleuten-Zeit",
        "standard": "Hawaii-Aleuten-Normalzeit",
        "daylight": "Hawaii-Aleuten-Sommerzeit"
      }
    },
    "Hong_Kong": {
      "long": {
        "generic": "Hongkong-Zeit",
        "standard": "Hongkong-Normalzeit",
        "daylight": "Hongkong-Sommerzeit"
      }
    },
    "Hovd": {
      "long": {
        "generic": "Chowd-Zeit",
        "standard": "Chowd-Normalzeit",
        "daylight": "Chowd-Sommerzeit"
      }
    },
    "India": {
      "long": {
        "standard": "Indische Zeit"
      }
    },
    "Indian_Ocean": {

```

```
        "long": {
          "standard": "Indischer Ozean-Zeit"
        }
      },
      "Indochina": {
        "long": {
          "standard": "Indochina-Zeit"
        }
      },
      "Indonesia_Central": {
        "long": {
          "standard": "Zentralindonesische Zeit"
        }
      },
      "Indonesia_Eastern": {
        "long": {
          "standard": "Ostindonesische Zeit"
        }
      },
      "Indonesia_Western": {
        "long": {
          "standard": "Westindonesische Zeit"
        }
      },
      "Iran": {
        "long": {
          "generic": "Iranische Zeit",
          "standard": "Iranische Normalzeit",
          "daylight": "Iranische Sommerzeit"
        }
      },
      "Irkutsk": {
        "long": {
          "generic": "Irkutsk-Zeit",
          "standard": "Irkutsk-Normalzeit",
          "daylight": "Irkutsk-Sommerzeit"
        }
      },
      "Israel": {
        "long": {
          "generic": "Israelische Zeit",
          "standard": "Israelische Normalzeit",
          "daylight": "Israelische Sommerzeit"
        }
      },
      "Japan": {
        "long": {
          "generic": "Japanische Zeit",
          "standard": "Japanische Normalzeit",
          "daylight": "Japanische Sommerzeit"
        }
      },
      "Kamchatka": {
        "long": {
          "generic": "Kamtschatka-Zeit",
          "standard": "Kamtschatka-Normalzeit",
          "daylight": "Kamtschatka-Sommerzeit"
        }
      }
    }
  }
}
```

```

    }
  },
  "Kazakhstan_Eastern": {
    "long": {
      "standard": "Ostkasachische Zeit"
    }
  },
  "Kazakhstan_Western": {
    "long": {
      "standard": "Westkasachische Zeit"
    }
  },
  "Korea": {
    "long": {
      "generic": "Koreanische Zeit",
      "standard": "Koreanische Normalzeit",
      "daylight": "Koreanische Sommerzeit"
    }
  },
  "Kosrae": {
    "long": {
      "standard": "Kosrae-Zeit"
    }
  },
  "Krasnoyarsk": {
    "long": {
      "generic": "Krasnojarsk-Zeit",
      "standard": "Krasnojarsk-Normalzeit",
      "daylight": "Krasnojarsk-Sommerzeit"
    }
  },
  "Kyrgystan": {
    "long": {
      "standard": "Kirgisistan-Zeit"
    }
  },
  "Lanka": {
    "long": {
      "standard": "Sri-Lanka-Zeit"
    }
  },
  "Line_Islands": {
    "long": {
      "standard": "Linieninseln-Zeit"
    }
  },
  "Lord_Howe": {
    "long": {
      "generic": "Lord-Howe-Zeit",
      "standard": "Lord-Howe-Normalzeit",
      "daylight": "Lord-Howe-Sommerzeit"
    }
  },
  "Macau": {
    "long": {
      "generic": "Macau-Zeit",
      "standard": "Macau-Normalzeit",

```

```

        "daylight": "Macau-Sommerzeit"
    },
    },
    "Macquarie": {
        "long": {
            "standard": "Macquarieinsel-Zeit"
        }
    },
    "Magadan": {
        "long": {
            "generic": "Magadan-Zeit",
            "standard": "Magadan-Normalzeit",
            "daylight": "Magadan-Sommerzeit"
        }
    },
    "Malaysia": {
        "long": {
            "standard": "Malaysische Zeit"
        }
    },
    "Maldives": {
        "long": {
            "standard": "Malediven-Zeit"
        }
    },
    "Marquesas": {
        "long": {
            "standard": "Marquesas-Zeit"
        }
    },
    "Marshall_Islands": {
        "long": {
            "standard": "Marshallinseln-Zeit"
        }
    },
    "Mauritius": {
        "long": {
            "generic": "Mauritius-Zeit",
            "standard": "Mauritius-Normalzeit",
            "daylight": "Mauritius-Sommerzeit"
        }
    },
    "Mawson": {
        "long": {
            "standard": "Mawson-Zeit"
        }
    },
    "Mexico_Northwest": {
        "long": {
            "generic": "Mexiko Nordwestliche Zone-Zeit",
            "standard": "Mexiko Nordwestliche Zone-Normalzeit",
            "daylight": "Mexiko Nordwestliche Zone-Sommerzeit"
        }
    },
    "Mexico_Pacific": {
        "long": {
            "generic": "Mexiko Pazifikzone-Zeit",

```

```

        "standard": "Mexiko Pazifikzone-Normalzeit",
        "daylight": "Mexiko Pazifikzone-Sommerzeit"
    },
    "Mongolia": {
        "long": {
            "generic": "Ulaanbaatar-Zeit",
            "standard": "Ulaanbaatar-Normalzeit",
            "daylight": "Ulaanbaatar-Sommerzeit"
        }
    },
    "Moscow": {
        "long": {
            "generic": "Moskauer Zeit",
            "standard": "Moskauer Normalzeit",
            "daylight": "Moskauer Sommerzeit"
        }
    },
    "Myanmar": {
        "long": {
            "standard": "Myanmar-Zeit"
        }
    },
    "Nauru": {
        "long": {
            "standard": "Nauru-Zeit"
        }
    },
    "Nepal": {
        "long": {
            "standard": "Nepalesische Zeit"
        }
    },
    "New_Caledonia": {
        "long": {
            "generic": "Neukaledonische Zeit",
            "standard": "Neukaledonische Normalzeit",
            "daylight": "Neukaledonische Sommerzeit"
        }
    },
    "New_Zealand": {
        "long": {
            "generic": "Neuseeland-Zeit",
            "standard": "Neuseeland-Normalzeit",
            "daylight": "Neuseeland-Sommerzeit"
        }
    },
    "Newfoundland": {
        "long": {
            "generic": "Neufundland-Zeit",
            "standard": "Neufundland-Normalzeit",
            "daylight": "Neufundland-Sommerzeit"
        }
    },
    "Niue": {
        "long": {
            "standard": "Niue-Zeit"
        }
    }

```

```

    }
  },
  "Norfolk": {
    "long": {
      "standard": "Norfolkinsel-Zeit"
    }
  },
  "Noronha": {
    "long": {
      "generic": "Fernando de Noronha-Zeit",
      "standard": "Fernando de Noronha-Normalzeit",
      "daylight": "Fernando de Noronha-Sommerzeit"
    }
  },
  "North_Mariana": {
    "long": {
      "standard": "Nördliche-Marianen-Zeit"
    }
  },
  "Novosibirsk": {
    "long": {
      "generic": "Nowosibirsk-Zeit",
      "standard": "Nowosibirsk-Normalzeit",
      "daylight": "Nowosibirsk-Sommerzeit"
    }
  },
  "Omsk": {
    "long": {
      "generic": "Omsk-Zeit",
      "standard": "Omsk-Normalzeit",
      "daylight": "Omsk-Sommerzeit"
    }
  },
  "Pakistan": {
    "long": {
      "generic": "Pakistanische Zeit",
      "standard": "Pakistanische Normalzeit",
      "daylight": "Pakistanische Sommerzeit"
    }
  },
  "Palau": {
    "long": {
      "standard": "Palau-Zeit"
    }
  },
  "Papua_New_Guinea": {
    "long": {
      "standard": "Papua-Neuguinea-Zeit"
    }
  },
  "Paraguay": {
    "long": {
      "generic": "Paraguayische Zeit",
      "standard": "Paraguayische Normalzeit",
      "daylight": "Paraguayische Sommerzeit"
    }
  },
},

```

```
"Peru": {
  "long": {
    "generic": "Peruanische Zeit",
    "standard": "Peruanische Normalzeit",
    "daylight": "Peruanische Sommerzeit"
  }
},
"Philippines": {
  "long": {
    "generic": "Philippinische Zeit",
    "standard": "Philippinische Normalzeit",
    "daylight": "Philippinische Sommerzeit"
  }
},
"Phoenix_Islands": {
  "long": {
    "standard": "Phoenixinseln-Zeit"
  }
},
"Pierre_Miquelon": {
  "long": {
    "generic": "Saint-Pierre-und-Miquelon-Zeit",
    "standard": "Saint-Pierre-und-Miquelon-Normalzeit",
    "daylight": "Saint-Pierre-und-Miquelon-Sommerzeit"
  }
},
"Pitcairn": {
  "long": {
    "standard": "Pitcairnsinseln-Zeit"
  }
},
"Ponape": {
  "long": {
    "standard": "Ponape-Zeit"
  }
},
"Pyongyang": {
  "long": {
    "standard": "Pjöngjang-Zeit"
  }
},
"Qyzylorda": {
  "long": {
    "generic": "Quysylorda-Zeit",
    "standard": "Quysylorda-Normalzeit",
    "daylight": "Qysylorda-Sommerzeit"
  }
},
"Reunion": {
  "long": {
    "standard": "Réunion-Zeit"
  }
},
"Rothera": {
  "long": {
    "standard": "Rothera-Zeit"
  }
}
```

```
    },
    "Sakhalin": {
      "long": {
        "generic": "Sachalin-Zeit",
        "standard": "Sachalin-Normalzeit",
        "daylight": "Sachalin-Sommerzeit"
      }
    },
    "Samara": {
      "long": {
        "generic": "Samara-Zeit",
        "standard": "Samara-Normalzeit",
        "daylight": "Samara-Sommerzeit"
      }
    },
    "Samoa": {
      "long": {
        "generic": "Samoa-Zeit",
        "standard": "Samoa-Normalzeit",
        "daylight": "Samoa-Sommerzeit"
      }
    },
    "Seychelles": {
      "long": {
        "standard": "Seychellen-Zeit"
      }
    },
    "Singapore": {
      "long": {
        "standard": "Singapur-Zeit"
      }
    },
    "Solomon": {
      "long": {
        "standard": "Salomoninseln-Zeit"
      }
    },
    "South_Georgia": {
      "long": {
        "standard": "Südgeorgische Zeit"
      }
    },
    "Suriname": {
      "long": {
        "standard": "Suriname-Zeit"
      }
    },
    "Syowa": {
      "long": {
        "standard": "Syowa-Zeit"
      }
    },
    "Tahiti": {
      "long": {
        "standard": "Tahiti-Zeit"
      }
    }
  },
}
```



```
"Taipei": {
  "long": {
    "generic": "Taipeh-Zeit",
    "standard": "Taipeh-Normalzeit",
    "daylight": "Taipeh-Sommerzeit"
  }
},
"Tajikistan": {
  "long": {
    "standard": "Tadschikistan-Zeit"
  }
},
"Tokelau": {
  "long": {
    "standard": "Tokelau-Zeit"
  }
},
"Tonga": {
  "long": {
    "generic": "Tonganische Zeit",
    "standard": "Tonganische Normalzeit",
    "daylight": "Tonganische Sommerzeit"
  }
},
"Truk": {
  "long": {
    "standard": "Chuuk-Zeit"
  }
},
"Turkmenistan": {
  "long": {
    "generic": "Turkmenistan-Zeit",
    "standard": "Turkmenistan-Normalzeit",
    "daylight": "Turkmenistan-Sommerzeit"
  }
},
"Tuvalu": {
  "long": {
    "standard": "Tuvalu-Zeit"
  }
},
"Uruguay": {
  "long": {
    "generic": "Uruguayische Zeit",
    "standard": "Uruguayische Normalzeit",
    "daylight": "Uruguayische Sommerzeit"
  }
},
"Uzbekistan": {
  "long": {
    "generic": "Usbekistan-Zeit",
    "standard": "Usbekistan-Normalzeit",
    "daylight": "Usbekistan-Sommerzeit"
  }
},
"Vanuatu": {
  "long": {
```

```

    "generic": "Vanuatu-Zeit",
    "standard": "Vanuatu-Normalzeit",
    "daylight": "Vanuatu-Sommerzeit"
  },
  "Venezuela": {
    "long": {
      "standard": "Venezuela-Zeit"
    }
  },
  "Vladivostok": {
    "long": {
      "generic": "Wladiwostok-Zeit",
      "standard": "Wladiwostok-Normalzeit",
      "daylight": "Wladiwostok-Sommerzeit"
    }
  },
  "Volgograd": {
    "long": {
      "generic": "Wolgograd-Zeit",
      "standard": "Wolgograd-Normalzeit",
      "daylight": "Wolgograd-Sommerzeit"
    }
  },
  "Vostok": {
    "long": {
      "standard": "Wostok-Zeit"
    }
  },
  "Wake": {
    "long": {
      "standard": "Wake-Insel-Zeit"
    }
  },
  "Wallis": {
    "long": {
      "standard": "Wallis-und-Futuna-Zeit"
    }
  },
  "Yakutsk": {
    "long": {
      "generic": "Jakutsk-Zeit",
      "standard": "Jakutsk-Normalzeit",
      "daylight": "Jakutsk-Sommerzeit"
    }
  },
  "Yekaterinburg": {
    "long": {
      "generic": "Jekaterinburg-Zeit",
      "standard": "Jekaterinburg-Normalzeit",
      "daylight": "Jekaterinburg-Sommerzeit"
    }
  }
}

```

```
}  
}
```

TIMEZONENAMES.JSX

```
{  
  "main";  
  {  
    "de";  
    {  
      "identity";  
      {  
        "version";  
        {  
          "_number";  
          "$Revision: 12879 $",  
          "_cldrVersion";  
          "30.0.3";  
        }  
        "language";  
        "de";  
      }  
      "dates";  
      {  
        "timeZoneNames";  
        {  
          "hourFormat";  
          "+HH:mm;-HH:mm",  
          "gmtFormat";  
          "GMT{0}",  
          "gmtZeroFormat";  
          "GMT",  
          "regionFormat";  
          "{0} Zeit",  
          "regionFormat-type-daylight";  
          "{0} Sommerzeit",  
          "regionFormat-type-standard";  
          "{0} Normalzeit",  
          "fallbackFormat";  
          "{1} ({0})",  
          "zone";  
          {  
            "America";  
            {  
              "Adak";  
              {  
                "exemplarCity";  
                "Adak";  
              }  
              "Anchorage";  
              {  
                "exemplarCity";  
                "Anchorage";  
              }  
            }  
            "Anguilla";  
            {
```

```
        "exemplarCity";
        "Anguilla";
    }
    "Antigua";
    {
        "exemplarCity";
        "Antigua";
    }
    "Araguaina";
    {
        "exemplarCity";
        "Araguaina";
    }
    "Argentina";
    {
        "Rio_Gallegos";
        {
            "exemplarCity";
            "Rio Gallegos";
        }
        "San_Juan";
        {
            "exemplarCity";
            "San Juan";
        }
        "Ushuaia";
        {
            "exemplarCity";
            "Ushuaia";
        }
        "La_Rioja";
        {
            "exemplarCity";
            "La Rioja";
        }
        "San_Luis";
        {
            "exemplarCity";
            "San Luis";
        }
        "Salta";
        {
            "exemplarCity";
            "Salta";
        }
        "Tucuman";
        {
            "exemplarCity";
            "Tucuman";
        }
    }
    "Aruba";
    {
        "exemplarCity";
        "Aruba";
    }
    "Asuncion";
```

```
{
    "exemplarCity";
    "Asunción";
}
"Bahia";
{
    "exemplarCity";
    "Bahia";
}
"Bahia_Banderas";
{
    "exemplarCity";
    "Bahia Banderas";
}
"Barbados";
{
    "exemplarCity";
    "Barbados";
}
"Belem";
{
    "exemplarCity";
    "Belem";
}
"Belize";
{
    "exemplarCity";
    "Belize";
}
"Blanc-Sablon";
{
    "exemplarCity";
    "Blanc-Sablon";
}
"Boa_Vista";
{
    "exemplarCity";
    "Boa Vista";
}
"Bogota";
{
    "exemplarCity";
    "Bogotá";
}
"Boise";
{
    "exemplarCity";
    "Boise";
}
"Buenos_Aires";
{
    "exemplarCity";
    "Buenos Aires";
}
"Cambridge_Bay";
{
    "exemplarCity";
```

```
        "Cambridge Bay";
    }
    "Campo_Grande";
    {
        "exemplarCity";
        "Campo Grande";
    }
    "Cancun";
    {
        "exemplarCity";
        "Cancún";
    }
    "Caracas";
    {
        "exemplarCity";
        "Caracas";
    }
    "Catamarca";
    {
        "exemplarCity";
        "Catamarca";
    }
    "Cayenne";
    {
        "exemplarCity";
        "Cayenne";
    }
    "Cayman";
    {
        "exemplarCity";
        "Kaimaninseln";
    }
    "Chicago";
    {
        "exemplarCity";
        "Chicago";
    }
    "Chihuahua";
    {
        "exemplarCity";
        "Chihuahua";
    }
    "Coral_Harbour";
    {
        "exemplarCity";
        "Atikokan";
    }
    "Cordoba";
    {
        "exemplarCity";
        "Córdoba";
    }
    "Costa_Rica";
    {
        "exemplarCity";
        "Costa Rica";
    }
}
```

```
"Creston";
{
  "exemplarCity";
  "Creston";
}
"Cuiaba";
{
  "exemplarCity";
  "Cuiaba";
}
"Curacao";
{
  "exemplarCity";
  "Curaçao";
}
"Danmarkshavn";
{
  "exemplarCity";
  "Danmarkshavn";
}
"Dawson";
{
  "exemplarCity";
  "Dawson";
}
"Dawson_Creek";
{
  "exemplarCity";
  "Dawson Creek";
}
"Denver";
{
  "exemplarCity";
  "Denver";
}
"Detroit";
{
  "exemplarCity";
  "Detroit";
}
"Dominica";
{
  "exemplarCity";
  "Dominica";
}
"Edmonton";
{
  "exemplarCity";
  "Edmonton";
}
"Eirunepe";
{
  "exemplarCity";
  "Eirunepe";
}
"El_Salvador";
{
```

```
        "exemplarCity";
        "El Salvador";
    }
    "Fort_Nelson";
    {
        "exemplarCity";
        "Fort Nelson";
    }
    "Fortaleza";
    {
        "exemplarCity";
        "Fortaleza";
    }
    "Glace_Bay";
    {
        "exemplarCity";
        "Glace Bay";
    }
    "Godthab";
    {
        "exemplarCity";
        "Nuuk";
    }
    "Goose_Bay";
    {
        "exemplarCity";
        "Goose Bay";
    }
    "Grand_Turk";
    {
        "exemplarCity";
        "Grand Turk";
    }
    "Grenada";
    {
        "exemplarCity";
        "Grenada";
    }
    "Guadeloupe";
    {
        "exemplarCity";
        "Guadeloupe";
    }
    "Guatemala";
    {
        "exemplarCity";
        "Guatemala";
    }
    "Guayaquil";
    {
        "exemplarCity";
        "Guayaquil";
    }
    "Guyana";
    {
        "exemplarCity";
        "Guyana";
    }
}
```



```
}
"Halifax";
{
  "exemplarCity";
  "Halifax";
}
"Havana";
{
  "exemplarCity";
  "Havanna";
}
"Hermosillo";
{
  "exemplarCity";
  "Hermosillo";
}
"Indiana";
{
  "Vincennes";
  {
    "exemplarCity";
    "Vincennes, Indiana";
  }
  "Petersburg";
  {
    "exemplarCity";
    "Petersburg, Indiana";
  }
  "Tell_City";
  {
    "exemplarCity";
    "Tell City, Indiana";
  }
  "Knox";
  {
    "exemplarCity";
    "Knox, Indiana";
  }
  "Winamac";
  {
    "exemplarCity";
    "Winamac, Indiana";
  }
  "Marengo";
  {
    "exemplarCity";
    "Marengo, Indiana";
  }
  "Vevay";
  {
    "exemplarCity";
    "Vevay, Indiana";
  }
}
"Indianapolis";
{
  "exemplarCity";
```

```
        "Indianapolis";
    }
    "Inuvik";
    {
        "exemplarCity";
        "Inuvik";
    }
    "Iqaluit";
    {
        "exemplarCity";
        "Iqaluit";
    }
    "Jamaica";
    {
        "exemplarCity";
        "Jamaika";
    }
    "Jujuy";
    {
        "exemplarCity";
        "Jujuy";
    }
    "Juneau";
    {
        "exemplarCity";
        "Juneau";
    }
    "Kentucky";
    {
        "Monticello";
        {
            "exemplarCity";
            "Monticello, Kentucky";
        }
    }
    "Kralendijk";
    {
        "exemplarCity";
        "Kralendijk";
    }
    "La_Paz";
    {
        "exemplarCity";
        "La Paz";
    }
    "Lima";
    {
        "exemplarCity";
        "Lima";
    }
    "Los_Angeles";
    {
        "exemplarCity";
        "Los Angeles";
    }
    "Louisville";
    {
```

```
        "exemplarCity";
        "Louisville";
    }
    "Lower_Princes";
    {
        "exemplarCity";
        "Lower Prince's Quarter";
    }
    "Maceio";
    {
        "exemplarCity";
        "Maceio";
    }
    "Managua";
    {
        "exemplarCity";
        "Managua";
    }
    "Manaus";
    {
        "exemplarCity";
        "Manaus";
    }
    "Marigot";
    {
        "exemplarCity";
        "Marigot";
    }
    "Martinique";
    {
        "exemplarCity";
        "Martinique";
    }
    "Matamoros";
    {
        "exemplarCity";
        "Matamoros";
    }
    "Mazatlan";
    {
        "exemplarCity";
        "Mazatlan";
    }
    "Mendoza";
    {
        "exemplarCity";
        "Mendoza";
    }
    "Menominee";
    {
        "exemplarCity";
        "Menominee";
    }
    "Merida";
    {
        "exemplarCity";
        "Merida";
    }
}
```

```
}
"Metlakatla";
{
  "exemplarCity";
  "Metlakatla";
}
"Mexico_City";
{
  "exemplarCity";
  "Mexiko-Stadt";
}
"Miquelon";
{
  "exemplarCity";
  "Miquelon";
}
"Moncton";
{
  "exemplarCity";
  "Moncton";
}
"Monterrey";
{
  "exemplarCity";
  "Monterrey";
}
"Montevideo";
{
  "exemplarCity";
  "Montevideo";
}
"Montserrat";
{
  "exemplarCity";
  "Montserrat";
}
"Nassau";
{
  "exemplarCity";
  "Nassau";
}
"New_York";
{
  "exemplarCity";
  "New York";
}
"Nipigon";
{
  "exemplarCity";
  "Nipigon";
}
"Nome";
{
  "exemplarCity";
  "Nome";
}
"Noronha";
```

```
{
  "exemplarCity";
  "Noronha";
}
"North_Dakota";
{
  "Beulah";
  {
    "exemplarCity";
    "Beulah, North Dakota";
  }
  "New_Salem";
  {
    "exemplarCity";
    "New Salem, North Dakota";
  }
  "Center";
  {
    "exemplarCity";
    "Center, North Dakota";
  }
}
"Ojinaga";
{
  "exemplarCity";
  "Ojinaga";
}
"Panama";
{
  "exemplarCity";
  "Panama";
}
"Pangnirtung";
{
  "exemplarCity";
  "Pangnirtung";
}
"Paramaribo";
{
  "exemplarCity";
  "Paramaribo";
}
"Phoenix";
{
  "exemplarCity";
  "Phoenix";
}
"Port-au-Prince";
{
  "exemplarCity";
  "Port-au-Prince";
}
"Port_of_Spain";
{
  "exemplarCity";
  "Port of Spain";
}
}
```

```
"Porto_Velho";
{
  "exemplarCity";
  "Porto Velho";
}
"Puerto_Rico";
{
  "exemplarCity";
  "Puerto Rico";
}
"Rainy_River";
{
  "exemplarCity";
  "Rainy River";
}
"Rankin_Inlet";
{
  "exemplarCity";
  "Rankin Inlet";
}
"Recife";
{
  "exemplarCity";
  "Recife";
}
"Regina";
{
  "exemplarCity";
  "Regina";
}
"Resolute";
{
  "exemplarCity";
  "Resolute";
}
"Rio_Branco";
{
  "exemplarCity";
  "Rio Branco";
}
"Santa_Isabel";
{
  "exemplarCity";
  "Santa Isabel";
}
"Santarem";
{
  "exemplarCity";
  "Santarem";
}
"Santiago";
{
  "exemplarCity";
  "Santiago";
}
"Santo_Domingo";
{
```

```
        "exemplarCity";
        "Santo Domingo";
    }
    "Sao_Paulo";
    {
        "exemplarCity";
        "São Paulo";
    }
    "Scoresbysund";
    {
        "exemplarCity";
        "Ittoqqortoormiit";
    }
    "Sitka";
    {
        "exemplarCity";
        "Sitka";
    }
    "St_Barthelemy";
    {
        "exemplarCity";
        "Saint-Barthélemy";
    }
    "St_Johns";
    {
        "exemplarCity";
        "St. John's";
    }
    "St_Kitts";
    {
        "exemplarCity";
        "St. Kitts";
    }
    "St_Lucia";
    {
        "exemplarCity";
        "St. Lucia";
    }
    "St_Thomas";
    {
        "exemplarCity";
        "St. Thomas";
    }
    "St_Vincent";
    {
        "exemplarCity";
        "St. Vincent";
    }
    "Swift_Current";
    {
        "exemplarCity";
        "Swift Current";
    }
    "Tegucigalpa";
    {
        "exemplarCity";
        "Tegucigalpa";
    }
}
```

```
}
  "Thule";
  {
    "exemplarCity";
    "Thule";
  }
  "Thunder_Bay";
  {
    "exemplarCity";
    "Thunder Bay";
  }
  "Tijuana";
  {
    "exemplarCity";
    "Tijuana";
  }
  "Toronto";
  {
    "exemplarCity";
    "Toronto";
  }
  "Tortola";
  {
    "exemplarCity";
    "Tortola";
  }
  "Vancouver";
  {
    "exemplarCity";
    "Vancouver";
  }
  "Whitehorse";
  {
    "exemplarCity";
    "Whitehorse";
  }
  "Winnipeg";
  {
    "exemplarCity";
    "Winnipeg";
  }
  "Yakutat";
  {
    "exemplarCity";
    "Yakutat";
  }
  "Yellowknife";
  {
    "exemplarCity";
    "Yellowknife";
  }
}
"Atlantic";
{
  "Azores";
  {
    "exemplarCity";
```



```

        "Azoren";
    }
    "Bermuda";
    {
        "exemplarCity";
        "Bermudas";
    }
    "Canary";
    {
        "exemplarCity";
        "Kanaren";
    }
    "Cape_Verde";
    {
        "exemplarCity";
        "Cabo Verde";
    }
    "Faeroe";
    {
        "exemplarCity";
        "Färöer";
    }
    "Madeira";
    {
        "exemplarCity";
        "Madeira";
    }
    "Reykjavik";
    {
        "exemplarCity";
        "Reykjavík";
    }
    "South_Georgia";
    {
        "exemplarCity";
        "Südgeorgien";
    }
    "St_Helena";
    {
        "exemplarCity";
        "St. Helena";
    }
    "Stanley";
    {
        "exemplarCity";
        "Stanley";
    }
}
"Europe";
{
    "Amsterdam";
    {
        "exemplarCity";
        "Amsterdam";
    }
    "Andorra";
    {

```

```
        "exemplarCity";
        "Andorra";
    }
    "Astrakhan";
    {
        "exemplarCity";
        "Astrachan";
    }
    "Athens";
    {
        "exemplarCity";
        "Athen";
    }
    "Belgrade";
    {
        "exemplarCity";
        "Belgrad";
    }
    "Berlin";
    {
        "exemplarCity";
        "Berlin";
    }
    "Bratislava";
    {
        "exemplarCity";
        "Bratislava";
    }
    "Brussels";
    {
        "exemplarCity";
        "Brüssel";
    }
    "Bucharest";
    {
        "exemplarCity";
        "Bukarest";
    }
    "Budapest";
    {
        "exemplarCity";
        "Budapest";
    }
    "Busingen";
    {
        "exemplarCity";
        "Büsingen";
    }
    "Chisinau";
    {
        "exemplarCity";
        "Kischinau";
    }
    "Copenhagen";
    {
        "exemplarCity";
        "Kopenhagen";
    }
```

```
}
"Dublin";
{
  "long";
  {
    "daylight";
    "Irische Sommerzeit";
  }
  "exemplarCity";
  "Dublin";
}
"Gibraltar";
{
  "exemplarCity";
  "Gibraltar";
}
"Guernsey";
{
  "exemplarCity";
  "Guernsey";
}
"Helsinki";
{
  "exemplarCity";
  "Helsinki";
}
"Isle_of_Man";
{
  "exemplarCity";
  "Isle of Man";
}
"Istanbul";
{
  "exemplarCity";
  "Istanbul";
}
"Jersey";
{
  "exemplarCity";
  "Jersey";
}
"Kaliningrad";
{
  "exemplarCity";
  "Kaliningrad";
}
"Kiev";
{
  "exemplarCity";
  "Kiew";
}
"Kirov";
{
  "exemplarCity";
  "Kirow";
}
"Lisbon";
```

```
{
  "exemplarCity";
  "Lissabon";
}
"Ljubljana";
{
  "exemplarCity";
  "Ljubljana";
}
"London";
{
  "long";
  {
    "daylight";
    "Britische Sommerzeit";
  }
  "exemplarCity";
  "London";
}
"Luxembourg";
{
  "exemplarCity";
  "Luxemburg";
}
"Madrid";
{
  "exemplarCity";
  "Madrid";
}
"Malta";
{
  "exemplarCity";
  "Malta";
}
"Mariehamn";
{
  "exemplarCity";
  "Mariehamn";
}
"Minsk";
{
  "exemplarCity";
  "Minsk";
}
"Monaco";
{
  "exemplarCity";
  "Monaco";
}
"Moscow";
{
  "exemplarCity";
  "Moskau";
}
"Oslo";
{
  "exemplarCity";
```

```
        "Oslo";
    }
    "Paris";
    {
        "exemplarCity";
        "Paris";
    }
    "Podgorica";
    {
        "exemplarCity";
        "Podgorica";
    }
    "Prague";
    {
        "exemplarCity";
        "Prag";
    }
    "Riga";
    {
        "exemplarCity";
        "Riga";
    }
    "Rome";
    {
        "exemplarCity";
        "Rom";
    }
    "Samara";
    {
        "exemplarCity";
        "Samara";
    }
    "San_Marino";
    {
        "exemplarCity";
        "San Marino";
    }
    "Sarajevo";
    {
        "exemplarCity";
        "Sarajevo";
    }
    "Simferopol";
    {
        "exemplarCity";
        "Simferopol";
    }
    "Skopje";
    {
        "exemplarCity";
        "Skopje";
    }
    "Sofia";
    {
        "exemplarCity";
        "Sofia";
    }
}
```

```
"Stockholm";
{
  "exemplarCity";
  "Stockholm";
}
"Tallinn";
{
  "exemplarCity";
  "Tallinn";
}
"Tirane";
{
  "exemplarCity";
  "Tirana";
}
"Ulyanovsk";
{
  "exemplarCity";
  "Uljanowsk";
}
"Uzhgorod";
{
  "exemplarCity";
  "Uschgorod";
}
"Vaduz";
{
  "exemplarCity";
  "Vaduz";
}
"Vatican";
{
  "exemplarCity";
  "Vatikan";
}
"Vienna";
{
  "exemplarCity";
  "Wien";
}
"Vilnius";
{
  "exemplarCity";
  "Vilnius";
}
"Volgograd";
{
  "exemplarCity";
  "Wolgograd";
}
"Warsaw";
{
  "exemplarCity";
  "Warschau";
}
"Zagreb";
{
```

```
        "exemplarCity";
        "Zagreb";
    }
    "Zaporozhye";
    {
        "exemplarCity";
        "Saporischja";
    }
    "Zurich";
    {
        "exemplarCity";
        "Zürich";
    }
}
"Africa";
{
    "Abidjan";
    {
        "exemplarCity";
        "Abidjan";
    }
    "Accra";
    {
        "exemplarCity";
        "Accra";
    }
    "Addis_Ababa";
    {
        "exemplarCity";
        "Addis Abeba";
    }
    "Algiers";
    {
        "exemplarCity";
        "Algier";
    }
    "Asmera";
    {
        "exemplarCity";
        "Asmara";
    }
    "Bamako";
    {
        "exemplarCity";
        "Bamako";
    }
    "Bangui";
    {
        "exemplarCity";
        "Bangui";
    }
    "Banjul";
    {
        "exemplarCity";
        "Banjul";
    }
    "Bissau";
```

```
{
  "exemplarCity";
  "Bissau";
}
"Blantyre";
{
  "exemplarCity";
  "Blantyre";
}
"Brazzaville";
{
  "exemplarCity";
  "Brazzaville";
}
"Bujumbura";
{
  "exemplarCity";
  "Bujumbura";
}
"Cairo";
{
  "exemplarCity";
  "Kairo";
}
"Casablanca";
{
  "exemplarCity";
  "Casablanca";
}
"Ceuta";
{
  "exemplarCity";
  "Ceuta";
}
"Conakry";
{
  "exemplarCity";
  "Conakry";
}
"Dakar";
{
  "exemplarCity";
  "Dakar";
}
"Dar_es_Salaam";
{
  "exemplarCity";
  "Daressalam";
}
"Djibouti";
{
  "exemplarCity";
  "Dschibuti";
}
"Douala";
{
  "exemplarCity";
```



```
        "Douala";
    }
    "El_Aaiun";
    {
        "exemplarCity";
        "El Aaiún";
    }
    "Freetown";
    {
        "exemplarCity";
        "Freetown";
    }
    "Gaborone";
    {
        "exemplarCity";
        "Gaborone";
    }
    "Harare";
    {
        "exemplarCity";
        "Harare";
    }
    "Johannesburg";
    {
        "exemplarCity";
        "Johannesburg";
    }
    "Juba";
    {
        "exemplarCity";
        "Juba";
    }
    "Kampala";
    {
        "exemplarCity";
        "Kampala";
    }
    "Khartoum";
    {
        "exemplarCity";
        "Khartum";
    }
    "Kigali";
    {
        "exemplarCity";
        "Kigali";
    }
    "Kinshasa";
    {
        "exemplarCity";
        "Kinshasa";
    }
    "Lagos";
    {
        "exemplarCity";
        "Lagos";
    }
}
```

```
"Libreville";
{
  "exemplarCity";
  "Libreville";
}
"Lome";
{
  "exemplarCity";
  "Lomé";
}
"Luanda";
{
  "exemplarCity";
  "Luanda";
}
"Lubumbashi";
{
  "exemplarCity";
  "Lubumbashi";
}
"Lusaka";
{
  "exemplarCity";
  "Lusaka";
}
"Malabo";
{
  "exemplarCity";
  "Malabo";
}
"Maputo";
{
  "exemplarCity";
  "Maputo";
}
"Maseru";
{
  "exemplarCity";
  "Maseru";
}
"Mbabane";
{
  "exemplarCity";
  "Mbabane";
}
"Mogadishu";
{
  "exemplarCity";
  "Mogadischu";
}
"Monrovia";
{
  "exemplarCity";
  "Monrovia";
}
"Nairobi";
{
```

```
        "exemplarCity";
        "Nairobi";
    }
    "Ndjamena";
    {
        "exemplarCity";
        "N'Djamena";
    }
    "Niamey";
    {
        "exemplarCity";
        "Niamey";
    }
    "Nouakchott";
    {
        "exemplarCity";
        "Nouakchott";
    }
    "Ouagadougou";
    {
        "exemplarCity";
        "Ouagadougou";
    }
    "Porto-Novo";
    {
        "exemplarCity";
        "Porto Novo";
    }
    "Sao_Tome";
    {
        "exemplarCity";
        "São Tomé";
    }
    "Tripoli";
    {
        "exemplarCity";
        "Tripolis";
    }
    "Tunis";
    {
        "exemplarCity";
        "Tunis";
    }
    "Windhoek";
    {
        "exemplarCity";
        "Windhoek";
    }
}
"Asia";
{
    "Aden";
    {
        "exemplarCity";
        "Aden";
    }
    "Almaty";
```

```
{
    "exemplarCity";
    "Almaty";
}
"Amman";
{
    "exemplarCity";
    "Amman";
}
"Anadyr";
{
    "exemplarCity";
    "Anadyr";
}
"Aqtau";
{
    "exemplarCity";
    "Aqtau";
}
"Aqtobe";
{
    "exemplarCity";
    "Aktobe";
}
"Ashgabat";
{
    "exemplarCity";
    "Aşgabat";
}
"Baghdad";
{
    "exemplarCity";
    "Bagdad";
}
"Bahrain";
{
    "exemplarCity";
    "Bahrain";
}
"Baku";
{
    "exemplarCity";
    "Baku";
}
"Bangkok";
{
    "exemplarCity";
    "Bangkok";
}
"Barnaul";
{
    "exemplarCity";
    "Barnaul";
}
"Beirut";
{
    "exemplarCity";
```

```
        "Beirut";
    }
    "Bishkek";
    {
        "exemplarCity";
        "Bischkek";
    }
    "Brunei";
    {
        "exemplarCity";
        "Brunei";
    }
    "Calcutta";
    {
        "exemplarCity";
        "Kalkutta";
    }
    "Chita";
    {
        "exemplarCity";
        "Tschita";
    }
    "Choibalsan";
    {
        "exemplarCity";
        "Tschoibalsan";
    }
    "Colombo";
    {
        "exemplarCity";
        "Colombo";
    }
    "Damascus";
    {
        "exemplarCity";
        "Damaskus";
    }
    "Dhaka";
    {
        "exemplarCity";
        "Dhaka";
    }
    "Dili";
    {
        "exemplarCity";
        "Dili";
    }
    "Dubai";
    {
        "exemplarCity";
        "Dubai";
    }
    "Dushanbe";
    {
        "exemplarCity";
        "Duschanbe";
    }
}
```

```
"Gaza";
{
  "exemplarCity";
  "Gaza";
}
"Hebron";
{
  "exemplarCity";
  "Hebron";
}
"Hong_Kong";
{
  "exemplarCity";
  "Hongkong";
}
"Hovd";
{
  "exemplarCity";
  "Chowd";
}
"Irkutsk";
{
  "exemplarCity";
  "Irkutsk";
}
"Jakarta";
{
  "exemplarCity";
  "Jakarta";
}
"Jayapura";
{
  "exemplarCity";
  "Jayapura";
}
"Jerusalem";
{
  "exemplarCity";
  "Jerusalem";
}
"Kabul";
{
  "exemplarCity";
  "Kabul";
}
"Kamchatka";
{
  "exemplarCity";
  "Kamtschatka";
}
"Karachi";
{
  "exemplarCity";
  "Karatschi";
}
"Katmandu";
{
```

```
        "exemplarCity";
        "Kathmandu";
    }
    "Khandyga";
    {
        "exemplarCity";
        "Chandyga";
    }
    "Krasnoyarsk";
    {
        "exemplarCity";
        "Krasnojarsk";
    }
    "Kuala_Lumpur";
    {
        "exemplarCity";
        "Kuala Lumpur";
    }
    "Kuching";
    {
        "exemplarCity";
        "Kuching";
    }
    "Kuwait";
    {
        "exemplarCity";
        "Kuwait";
    }
    "Macau";
    {
        "exemplarCity";
        "Macao";
    }
    "Magadan";
    {
        "exemplarCity";
        "Magadan";
    }
    "Makassar";
    {
        "exemplarCity";
        "Makassar";
    }
    "Manila";
    {
        "exemplarCity";
        "Manila";
    }
    "Muscat";
    {
        "exemplarCity";
        "Maskat";
    }
    "Nicosia";
    {
        "exemplarCity";
        "Nikosia";
    }
}
```

```
}
"Novokuznetsk";
{
  "exemplarCity";
  "Nowokuznetsk";
}
"Novosibirsk";
{
  "exemplarCity";
  "Nowosibirsk";
}
"Omsk";
{
  "exemplarCity";
  "Omsk";
}
"Oral";
{
  "exemplarCity";
  "Oral";
}
"Phnom_Penh";
{
  "exemplarCity";
  "Phnom Penh";
}
"Pontianak";
{
  "exemplarCity";
  "Pontianak";
}
"Pyongyang";
{
  "exemplarCity";
  "Pjöngjang";
}
"Qatar";
{
  "exemplarCity";
  "Katar";
}
"Qyzylorda";
{
  "exemplarCity";
  "Qysylorda";
}
"Rangoon";
{
  "exemplarCity";
  "Rangun";
}
"Riyadh";
{
  "exemplarCity";
  "Riad";
}
"Saigon";
```



```
{
  "exemplarCity";
  "Ho-Chi-Minh-Stadt";
}
"Sakhalin";
{
  "exemplarCity";
  "Sachalin";
}
"Samarkand";
{
  "exemplarCity";
  "Samarkand";
}
"Seoul";
{
  "exemplarCity";
  "Seoul";
}
"Shanghai";
{
  "exemplarCity";
  "Shanghai";
}
"Singapore";
{
  "exemplarCity";
  "Singapur";
}
"Srednekolymsk";
{
  "exemplarCity";
  "Srednekolymsk";
}
"Taipei";
{
  "exemplarCity";
  "Taipeh";
}
"Tashkent";
{
  "exemplarCity";
  "Taschkent";
}
"Tbilisi";
{
  "exemplarCity";
  "Tiflis";
}
"Tehran";
{
  "exemplarCity";
  "Teheran";
}
"Thimphu";
{
  "exemplarCity";
```

```
        "Thimphu";
    }
    "Tokyo";
    {
        "exemplarCity";
        "Tokio";
    }
    "Tomsk";
    {
        "exemplarCity";
        "Tomsk";
    }
    "Ulaanbaatar";
    {
        "exemplarCity";
        "Ulaanbaatar";
    }
    "Urumqi";
    {
        "exemplarCity";
        "Ürümqi";
    }
    "Ust-Nera";
    {
        "exemplarCity";
        "Ust-Nera";
    }
    "Vientiane";
    {
        "exemplarCity";
        "Vientiane";
    }
    "Vladivostok";
    {
        "exemplarCity";
        "Wladiwostok";
    }
    "Yakutsk";
    {
        "exemplarCity";
        "Jakutsk";
    }
    "Yekaterinburg";
    {
        "exemplarCity";
        "Jekaterinburg";
    }
    "Yerevan";
    {
        "exemplarCity";
        "Eriwan";
    }
}
"Indian";
{
    "Antananarivo";
    {
```

```
        "exemplarCity";
        "Antananarivo";
    }
    "Chagos";
    {
        "exemplarCity";
        "Chagos";
    }
    "Christmas";
    {
        "exemplarCity";
        "Weihnachtsinsel";
    }
    "Cocos";
    {
        "exemplarCity";
        "Cocos";
    }
    "Comoro";
    {
        "exemplarCity";
        "Komoren";
    }
    "Kerguelen";
    {
        "exemplarCity";
        "Kerguelen";
    }
    "Mahe";
    {
        "exemplarCity";
        "Mahe";
    }
    "Maldives";
    {
        "exemplarCity";
        "Malediven";
    }
    "Mauritius";
    {
        "exemplarCity";
        "Mauritius";
    }
    "Mayotte";
    {
        "exemplarCity";
        "Mayotte";
    }
    "Reunion";
    {
        "exemplarCity";
        "Réunion";
    }
    }
    "Australia";
    {
        "Adelaide";
```

```
{
    "exemplarCity";
    "Adelaide";
}
"Brisbane";
{
    "exemplarCity";
    "Brisbane";
}
"Broken_Hill";
{
    "exemplarCity";
    "Broken Hill";
}
"Currie";
{
    "exemplarCity";
    "Currie";
}
"Darwin";
{
    "exemplarCity";
    "Darwin";
}
"Eucla";
{
    "exemplarCity";
    "Eucla";
}
"Hobart";
{
    "exemplarCity";
    "Hobart";
}
"Lindeman";
{
    "exemplarCity";
    "Lindeman";
}
"Lord_Howe";
{
    "exemplarCity";
    "Lord Howe";
}
"Melbourne";
{
    "exemplarCity";
    "Melbourne";
}
"Perth";
{
    "exemplarCity";
    "Perth";
}
"Sydney";
{
    "exemplarCity";
```

```
        "Sydney";
    }
}
"Pacific";
{
    "Apia";
    {
        "exemplarCity";
        "Apia";
    }
    "Auckland";
    {
        "exemplarCity";
        "Auckland";
    }
    "Bougainville";
    {
        "exemplarCity";
        "Bougainville";
    }
    "Chatham";
    {
        "exemplarCity";
        "Chatham";
    }
    "Easter";
    {
        "exemplarCity";
        "Osterinsel";
    }
    "Efate";
    {
        "exemplarCity";
        "Efate";
    }
    "Enderbury";
    {
        "exemplarCity";
        "Enderbury";
    }
    "Fakaofo";
    {
        "exemplarCity";
        "Fakaofo";
    }
    "Fiji";
    {
        "exemplarCity";
        "Fidschi";
    }
    "Funafuti";
    {
        "exemplarCity";
        "Funafuti";
    }
    "Galapagos";
    {
```

```
        "exemplarCity";
        "Galapagos";
    }
    "Gambier";
    {
        "exemplarCity";
        "Gambier";
    }
    "Guadalcanal";
    {
        "exemplarCity";
        "Guadalcanal";
    }
    "Guam";
    {
        "exemplarCity";
        "Guam";
    }
    "Honolulu";
    {
        "exemplarCity";
        "Honolulu";
    }
    "Johnston";
    {
        "exemplarCity";
        "Johnston";
    }
    "Kiritimati";
    {
        "exemplarCity";
        "Kiritimati";
    }
    "Kosrae";
    {
        "exemplarCity";
        "Kosrae";
    }
    "Kwajalein";
    {
        "exemplarCity";
        "Kwajalein";
    }
    "Majuro";
    {
        "exemplarCity";
        "Majuro";
    }
    "Marquesas";
    {
        "exemplarCity";
        "Marquesas";
    }
    "Midway";
    {
        "exemplarCity";
        "Midway";
    }
```

```
}
"Nauru";
{
  "exemplarCity";
  "Nauru";
}
"Niue";
{
  "exemplarCity";
  "Niue";
}
"Norfolk";
{
  "exemplarCity";
  "Norfolk";
}
"Noumea";
{
  "exemplarCity";
  "Noumea";
}
"Pago_Pago";
{
  "exemplarCity";
  "Pago Pago";
}
"Palau";
{
  "exemplarCity";
  "Palau";
}
"Pitcairn";
{
  "exemplarCity";
  "Pitcairn";
}
"Ponape";
{
  "exemplarCity";
  "Pohnpei";
}
"Port_Moresby";
{
  "exemplarCity";
  "Port Moresby";
}
"Rarotonga";
{
  "exemplarCity";
  "Rarotonga";
}
"Saipan";
{
  "exemplarCity";
  "Saipan";
}
"Tahiti";
```

```
{
    "exemplarCity";
    "Tahiti";
}
"Tarawa";
{
    "exemplarCity";
    "Tarawa";
}
"Tongatapu";
{
    "exemplarCity";
    "Tongatapu";
}
"Truk";
{
    "exemplarCity";
    "Chuuk";
}
"Wake";
{
    "exemplarCity";
    "Wake";
}
"Wallis";
{
    "exemplarCity";
    "Wallis";
}
}
"Arctic";
{
    "Longyearbyen";
    {
        "exemplarCity";
        "Longyearbyen";
    }
}
"Antarctica";
{
    "Casey";
    {
        "exemplarCity";
        "Casey";
    }
    "Davis";
    {
        "exemplarCity";
        "Davis";
    }
    "DumontDUrville";
    {
        "exemplarCity";
        "Dumont d'Urville";
    }
    "Macquarie";
    {
```



```
        "exemplarCity";
        "Macquarie";
    }
    "Mawson";
    {
        "exemplarCity";
        "Mawson";
    }
    "McMurdo";
    {
        "exemplarCity";
        "McMurdo";
    }
    "Palmer";
    {
        "exemplarCity";
        "Palmer";
    }
    "Rothera";
    {
        "exemplarCity";
        "Rothera";
    }
    "Syowa";
    {
        "exemplarCity";
        "Syowa";
    }
    "Troll";
    {
        "exemplarCity";
        "Troll";
    }
    "Vostok";
    {
        "exemplarCity";
        "Wostok";
    }
}
"Etc";
{
    "GMT";
    {
        "exemplarCity";
        "GMT";
    }
    "GMT1";
    {
        "exemplarCity";
        "GMT+1";
    }
    "GMT10";
    {
        "exemplarCity";
        "GMT+10";
    }
    "GMT11";
```

```
{
    "exemplarCity";
    "GMT+11";
}
"GMT12";
{
    "exemplarCity";
    "GMT+12";
}
"GMT2";
{
    "exemplarCity";
    "GMT+2";
}
"GMT3";
{
    "exemplarCity";
    "GMT+3";
}
"GMT4";
{
    "exemplarCity";
    "GMT+4";
}
"GMT5";
{
    "exemplarCity";
    "GMT+5";
}
"GMT6";
{
    "exemplarCity";
    "GMT+6";
}
"GMT7";
{
    "exemplarCity";
    "GMT+7";
}
"GMT8";
{
    "exemplarCity";
    "GMT+8";
}
"GMT9";
{
    "exemplarCity";
    "GMT+9";
}
"GMT-1";
{
    "exemplarCity";
    "GMT-1";
}
"GMT-10";
{
    "exemplarCity";
```

```
        "GMT-10";
    }
    "GMT-11";
    {
        "exemplarCity";
        "GMT-11";
    }
    "GMT-12";
    {
        "exemplarCity";
        "GMT-12";
    }
    "GMT-13";
    {
        "exemplarCity";
        "GMT-13";
    }
    "GMT-14";
    {
        "exemplarCity";
        "GMT-14";
    }
    "GMT-2";
    {
        "exemplarCity";
        "GMT-2";
    }
    "GMT-3";
    {
        "exemplarCity";
        "GMT-3";
    }
    "GMT-4";
    {
        "exemplarCity";
        "GMT-4";
    }
    "GMT-5";
    {
        "exemplarCity";
        "GMT-5";
    }
    "GMT-6";
    {
        "exemplarCity";
        "GMT-6";
    }
    "GMT-7";
    {
        "exemplarCity";
        "GMT-7";
    }
    "GMT-8";
    {
        "exemplarCity";
        "GMT-8";
    }
}
```

```

        "GMT-9";
        {
            "exemplarCity";
            "GMT-9";
        }
        "Unknown";
        {
            "exemplarCity";
            "Unbekannt";
        }
    }
    "metazone";
    {
        "Acre";
        {
            "long";
            {
                "generic";
                "Acre-Zeit",
                "standard";
                "Acre-Normalzeit",
                "daylight";
                "Acre-Sommerzeit";
            }
        }
        "Afghanistan";
        {
            "long";
            {
                "standard";
                "Afghanistan-Zeit";
            }
        }
        "Africa_Central";
        {
            "long";
            {
                "standard";
                "Zentralafrikanische Zeit";
            }
        }
        "Africa_Eastern";
        {
            "long";
            {
                "standard";
                "Ostafrikanische Zeit";
            }
        }
        "Africa_Southern";
        {
            "long";
            {
                "standard";
                "Südafrikanische Zeit";
            }
        }
    }

```

```
}
"Africa_Western";
{
  "long";
  {
    "generic";
    "Westafrikanische Zeit",
    "standard";
    "Westafrikanische Normalzeit",
    "daylight";
    "Westafrikanische Sommerzeit";
  }
}
"Alaska";
{
  "long";
  {
    "generic";
    "Alaska-Zeit",
    "standard";
    "Alaska-Normalzeit",
    "daylight";
    "Alaska-Sommerzeit";
  }
}
"Almaty";
{
  "long";
  {
    "generic";
    "Almaty-Zeit",
    "standard";
    "Almaty-Normalzeit",
    "daylight";
    "Almaty-Sommerzeit";
  }
}
"Amazon";
{
  "long";
  {
    "generic";
    "Amazonas-Zeit",
    "standard";
    "Amazonas-Normalzeit",
    "daylight";
    "Amazonas-Sommerzeit";
  }
}
"America_Central";
{
  "long";
  {
    "generic";
    "Nordamerikanische Inlandzeit",
    "standard";
    "Nordamerikanische Inland-Normalzeit",
```

```

        "daylight";
        "Nordamerikanische Inland-Sommerzeit";
    }
}
"America_Eastern";
{
    "long";
    {
        "generic";
        "Nordamerikanische Ostküstenzeit",
        "standard";
        "Nordamerikanische Ostküsten-Normalzeit",
        "daylight";
        "Nordamerikanische Ostküsten-Sommerzeit";
    }
}
"America_Mountain";
{
    "long";
    {
        "generic";
        "Rocky-Mountain-Zeit",
        "standard";
        "Rocky Mountain-Normalzeit",
        "daylight";
        "Rocky-Mountain-Sommerzeit";
    }
}
"America_Pacific";
{
    "long";
    {
        "generic";
        "Nordamerikanische Westküstenzeit",
        "standard";
        "Nordamerikanische Westküsten-Normalzeit",
        "daylight";
        "Nordamerikanische Westküsten-Sommerzeit";
    }
}
"Anadyr";
{
    "long";
    {
        "generic";
        "Anadyr Zeit",
        "standard";
        "Anadyr Normalzeit",
        "daylight";
        "Anadyr Sommerzeit";
    }
}
"Apia";
{
    "long";
    {
        "generic";

```

```

        "Apia-Zeit",
        "standard";
        "Apia-Normalzeit",
        "daylight";
        "Apia-Sommerzeit";
    }
}
"Aqtau";
{
    "long";
    {
        "generic";
        "Aqtau-Zeit",
        "standard";
        "Aqtau-Normalzeit",
        "daylight";
        "Aqtau-Sommerzeit";
    }
}
"Aqtobe";
{
    "long";
    {
        "generic";
        "Aqtöbe-Zeit",
        "standard";
        "Aqtöbe-Normalzeit",
        "daylight";
        "Aqtöbe-Sommerzeit";
    }
}
"Arabian";
{
    "long";
    {
        "generic";
        "Arabische Zeit",
        "standard";
        "Arabische Normalzeit",
        "daylight";
        "Arabische Sommerzeit";
    }
}
"Argentina";
{
    "long";
    {
        "generic";
        "Argentinische Zeit",
        "standard";
        "Argentinische Normalzeit",
        "daylight";
        "Argentinische Sommerzeit";
    }
}
"Argentina_Western";
{

```

```
        "long";
        {
            "generic";
            "Westargentinische Zeit",
            "standard";
            "Westargentinische Normalzeit",
            "daylight";
            "Westargentinische Sommerzeit";
        }
    }
    "Armenia";
    {
        "long";
        {
            "generic";
            "Armenische Zeit",
            "standard";
            "Armenische Normalzeit",
            "daylight";
            "Armenische Sommerzeit";
        }
    }
    "Atlantic";
    {
        "long";
        {
            "generic";
            "Atlantik-Zeit",
            "standard";
            "Atlantik-Normalzeit",
            "daylight";
            "Atlantik-Sommerzeit";
        }
    }
    "Australia_Central";
    {
        "long";
        {
            "generic";
            "Zentralaustralische Zeit",
            "standard";
            "Zentralaustralische Normalzeit",
            "daylight";
            "Zentralaustralische Sommerzeit";
        }
    }
    "Australia_CentralWestern";
    {
        "long";
        {
            "generic";
            "Zentral-/Westaustralische Zeit",
            "standard";
            "Zentral-/Westaustralische Normalzeit",
            "daylight";
            "Zentral-/Westaustralische Sommerzeit";
        }
    }
}
```



```
}
"Australia_Eastern";
{
  "long";
  {
    "generic";
    "Ostaustralische Zeit",
    "standard";
    "Ostaustralische Normalzeit",
    "daylight";
    "Ostaustralische Sommerzeit";
  }
}
"Australia_Western";
{
  "long";
  {
    "generic";
    "Westaustralische Zeit",
    "standard";
    "Westaustralische Normalzeit",
    "daylight";
    "Westaustralische Sommerzeit";
  }
}
"Azerbaijan";
{
  "long";
  {
    "generic";
    "Aserbajdschanische Zeit",
    "standard";
    "Aserbajdschanische Normalzeit",
    "daylight";
    "Aserbajdschanische Sommerzeit";
  }
}
"Azores";
{
  "long";
  {
    "generic";
    "Azoren-Zeit",
    "standard";
    "Azoren-Normalzeit",
    "daylight";
    "Azoren-Sommerzeit";
  }
}
"Bangladesh";
{
  "long";
  {
    "generic";
    "Bangladesch-Zeit",
    "standard";
    "Bangladesch-Normalzeit",
```

```

        "daylight";
        "Bangladesch-Sommerzeit";
    }
}
"Bhutan";
{
    "long";
    {
        "standard";
        "Bhutan-Zeit";
    }
}
"Bolivia";
{
    "long";
    {
        "standard";
        "Bolivianische Zeit";
    }
}
"Brasilia";
{
    "long";
    {
        "generic";
        "Brasília-Zeit",
        "standard";
        "Brasília-Normalzeit",
        "daylight";
        "Brasília-Sommerzeit";
    }
}
"Brunei";
{
    "long";
    {
        "standard";
        "Brunei-Zeit";
    }
}
"Cape_Verde";
{
    "long";
    {
        "generic";
        "Cabo-Verde-Zeit",
        "standard";
        "Cabo-Verde-Normalzeit",
        "daylight";
        "Cabo-Verde-Sommerzeit";
    }
}
"Casey";
{
    "long";
    {
        "standard";

```

```

        "Casey-Zeit";
    }
}
"Chamorro";
{
    "long";
    {
        "standard";
        "Chamorro-Zeit";
    }
}
"Chatham";
{
    "long";
    {
        "generic";
        "Chatham-Zeit",
        "standard";
        "Chatham-Normalzeit",
        "daylight";
        "Chatham-Sommerzeit";
    }
}
"Chile";
{
    "long";
    {
        "generic";
        "Chilenische Zeit",
        "standard";
        "Chilenische Normalzeit",
        "daylight";
        "Chilenische Sommerzeit";
    }
}
"China";
{
    "long";
    {
        "generic";
        "Chinesische Zeit",
        "standard";
        "Chinesische Normalzeit",
        "daylight";
        "Chinesische Sommerzeit";
    }
}
"Choibalsan";
{
    "long";
    {
        "generic";
        "Tschoibalsan-Zeit",
        "standard";
        "Tschoibalsan-Normalzeit",
        "daylight";
        "Tschoibalsan-Sommerzeit";
    }
}

```

```
    }  
  }  
  "Christmas";  
  {  
    "long";  
    {  
      "standard";  
      "Weihnachtsinsel-Zeit";  
    }  
  }  
  "Cocos";  
  {  
    "long";  
    {  
      "standard";  
      "Kokosinseln-Zeit";  
    }  
  }  
  "Colombia";  
  {  
    "long";  
    {  
      "generic";  
      "Kolumbianische Zeit",  
      "standard";  
      "Kolumbianische Normalzeit",  
      "daylight";  
      "Kolumbianische Sommerzeit";  
    }  
  }  
  "Cook";  
  {  
    "long";  
    {  
      "generic";  
      "Cookinseln-Zeit",  
      "standard";  
      "Cookinseln-Normalzeit",  
      "daylight";  
      "Cookinseln-Sommerzeit";  
    }  
  }  
  "Cuba";  
  {  
    "long";  
    {  
      "generic";  
      "Kubanische Zeit",  
      "standard";  
      "Kubanische Normalzeit",  
      "daylight";  
      "Kubanische Sommerzeit";  
    }  
  }  
  "Davis";  
  {  
    "long";
```

```
        {
            "standard";
            "Davis-Zeit";
        }
    }
    "DumontDUrville";
    {
        "long";
        {
            "standard";
            "Dumont-d'Urville-Zeit";
        }
    }
    "East_Timor";
    {
        "long";
        {
            "standard";
            "Osttimor-Zeit";
        }
    }
    "Easter";
    {
        "long";
        {
            "generic";
            "Osterinsel-Zeit",
            "standard";
            "Osterinsel-Normalzeit",
            "daylight";
            "Osterinsel-Sommerzeit";
        }
    }
    "Ecuador";
    {
        "long";
        {
            "standard";
            "Ecuadorianische Zeit";
        }
    }
    "Europe_Central";
    {
        "long";
        {
            "generic";
            "Mittleuropäische Zeit",
            "standard";
            "Mittleuropäische Normalzeit",
            "daylight";
            "Mittleuropäische Sommerzeit";
        }
        "short";
        {
            "generic";
            "MEZ",
            "standard";
        }
    }
}
```

```

        "MEZ",
        "daylight";
        "MESZ";
    }
}
"Europe_Eastern";
{
    "long";
    {
        "generic";
        "Osteuropäische Zeit",
        "standard";
        "Osteuropäische Normalzeit",
        "daylight";
        "Osteuropäische Sommerzeit";
    }
    "short";
    {
        "generic";
        "OEZ",
        "standard";
        "OEZ",
        "daylight";
        "OESZ";
    }
}
"Europe_Further_Eastern";
{
    "long";
    {
        "standard";
        "Kaliningrader Zeit";
    }
}
"Europe_Western";
{
    "long";
    {
        "generic";
        "Westeuropäische Zeit",
        "standard";
        "Westeuropäische Normalzeit",
        "daylight";
        "Westeuropäische Sommerzeit";
    }
    "short";
    {
        "generic";
        "WEZ",
        "standard";
        "WEZ",
        "daylight";
        "WESZ";
    }
}
"Falkland";
{

```

```

        "long";
        {
            "generic";
            "Falklandinseln-Zeit",
            "standard";
            "Falklandinseln-Normalzeit",
            "daylight";
            "Falklandinseln-Sommerzeit";
        }
    }
    "Fiji";
    {
        "long";
        {
            "generic";
            "Fidschi-Zeit",
            "standard";
            "Fidschi-Normalzeit",
            "daylight";
            "Fidschi-Sommerzeit";
        }
    }
    "French_Guiana";
    {
        "long";
        {
            "standard";
            "Französisch-Guayana-Zeit";
        }
    }
    "French_Southern";
    {
        "long";
        {
            "standard";
            "Französische Süd- und Antarktisgebiete-
Zeit";
        }
    }
    "Galapagos";
    {
        "long";
        {
            "standard";
            "Galapagos-Zeit";
        }
    }
    "Gambier";
    {
        "long";
        {
            "standard";
            "Gambier-Zeit";
        }
    }
    "Georgia";
    {

```

```
        "long";
        {
            "generic";
            "Georgische Zeit",
            "standard";
            "Georgische Normalzeit",
            "daylight";
            "Georgische Sommerzeit";
        }
    }
    "Gilbert_Islands";
    {
        "long";
        {
            "standard";
            "Gilbert-Inseln-Zeit";
        }
    }
    "GMT";
    {
        "long";
        {
            "standard";
            "Mittlere Greenwich-Zeit";
        }
    }
    "Greenland_Eastern";
    {
        "long";
        {
            "generic";
            "Ostgrönland-Zeit",
            "standard";
            "Ostgrönland-Normalzeit",
            "daylight";
            "Ostgrönland-Sommerzeit";
        }
    }
    "Greenland_Western";
    {
        "long";
        {
            "generic";
            "Westgrönland-Zeit",
            "standard";
            "Westgrönland-Normalzeit",
            "daylight";
            "Westgrönland-Sommerzeit";
        }
    }
    "Guam";
    {
        "long";
        {
            "standard";
            "Guam-Zeit";
        }
    }
}
```



```
}
"Gulf";
{
  "long";
  {
    "standard";
    "Golf-Zeit";
  }
}
"Guyana";
{
  "long";
  {
    "standard";
    "Guyana-Zeit";
  }
}
"Hawaii_Aleutian";
{
  "long";
  {
    "generic";
    "Hawaii-Aleuten-Zeit",
    "standard";
    "Hawaii-Aleuten-Normalzeit",
    "daylight";
    "Hawaii-Aleuten-Sommerzeit";
  }
}
"Hong_Kong";
{
  "long";
  {
    "generic";
    "Hongkong-Zeit",
    "standard";
    "Hongkong-Normalzeit",
    "daylight";
    "Hongkong-Sommerzeit";
  }
}
"Hovd";
{
  "long";
  {
    "generic";
    "Chowd-Zeit",
    "standard";
    "Chowd-Normalzeit",
    "daylight";
    "Chowd-Sommerzeit";
  }
}
"India";
{
  "long";
  {
```

```
        "standard";
        "Indische Zeit";
    }
}
"Indian_Ocean";
{
    "long";
    {
        "standard";
        "Indischer Ozean-Zeit";
    }
}
"Indochina";
{
    "long";
    {
        "standard";
        "Indochina-Zeit";
    }
}
"Indonesia_Central";
{
    "long";
    {
        "standard";
        "Zentralindonesische Zeit";
    }
}
"Indonesia_Eastern";
{
    "long";
    {
        "standard";
        "Ostindonesische Zeit";
    }
}
"Indonesia_Western";
{
    "long";
    {
        "standard";
        "Westindonesische Zeit";
    }
}
"Iran";
{
    "long";
    {
        "generic";
        "Iranische Zeit",
        "standard";
        "Iranische Normalzeit",
        "daylight";
        "Iranische Sommerzeit";
    }
}
"Irkutsk";
```

```
{
  "long";
  {
    "generic";
    "Irkutsk-Zeit",
    "standard";
    "Irkutsk-Normalzeit",
    "daylight";
    "Irkutsk-Sommerzeit";
  }
}
"Israel";
{
  "long";
  {
    "generic";
    "Israelische Zeit",
    "standard";
    "Israelische Normalzeit",
    "daylight";
    "Israelische Sommerzeit";
  }
}
"Japan";
{
  "long";
  {
    "generic";
    "Japanische Zeit",
    "standard";
    "Japanische Normalzeit",
    "daylight";
    "Japanische Sommerzeit";
  }
}
"Kamchatka";
{
  "long";
  {
    "generic";
    "Kamtschatka-Zeit",
    "standard";
    "Kamtschatka-Normalzeit",
    "daylight";
    "Kamtschatka-Sommerzeit";
  }
}
"Kazakhstan_Eastern";
{
  "long";
  {
    "standard";
    "Ostkasachische Zeit";
  }
}
"Kazakhstan_Western";
{
```

```
        "long";
        {
            "standard";
            "Westkasachische Zeit";
        }
    }
    "Korea";
    {
        "long";
        {
            "generic";
            "Koreanische Zeit",
            "standard";
            "Koreanische Normalzeit",
            "daylight";
            "Koreanische Sommerzeit";
        }
    }
    "Kosrae";
    {
        "long";
        {
            "standard";
            "Kosrae-Zeit";
        }
    }
    "Krasnojarsk";
    {
        "long";
        {
            "generic";
            "Krasnojarsk-Zeit",
            "standard";
            "Krasnojarsk-Normalzeit",
            "daylight";
            "Krasnojarsk-Sommerzeit";
        }
    }
    "Kyrgystan";
    {
        "long";
        {
            "standard";
            "Kirgisistan-Zeit";
        }
    }
    "Lanka";
    {
        "long";
        {
            "standard";
            "Sri-Lanka-Zeit";
        }
    }
    "Line_Islands";
    {
        "long";
```

```
        {
            "standard";
            "Linieninseln-Zeit";
        }
    }
    "Lord_Howe";
    {
        "long";
        {
            "generic";
            "Lord-Howe-Zeit",
            "standard";
            "Lord-Howe-Normalzeit",
            "daylight";
            "Lord-Howe-Sommerzeit";
        }
    }
    "Macau";
    {
        "long";
        {
            "generic";
            "Macau-Zeit",
            "standard";
            "Macau-Normalzeit",
            "daylight";
            "Macau-Sommerzeit";
        }
    }
    "Macquarie";
    {
        "long";
        {
            "standard";
            "Macquarieinsel-Zeit";
        }
    }
    "Magadan";
    {
        "long";
        {
            "generic";
            "Magadan-Zeit",
            "standard";
            "Magadan-Normalzeit",
            "daylight";
            "Magadan-Sommerzeit";
        }
    }
    "Malaysia";
    {
        "long";
        {
            "standard";
            "Malaysische Zeit";
        }
    }
}
```

```
"Maldives";
{
  "long";
  {
    "standard";
    "Malediven-Zeit";
  }
}
"Marquesas";
{
  "long";
  {
    "standard";
    "Marquesas-Zeit";
  }
}
"Marshall_Islands";
{
  "long";
  {
    "standard";
    "Marshallinseln-Zeit";
  }
}
"Mauritius";
{
  "long";
  {
    "generic";
    "Mauritius-Zeit",
    "standard";
    "Mauritius-Normalzeit",
    "daylight";
    "Mauritius-Sommerzeit";
  }
}
"Mawson";
{
  "long";
  {
    "standard";
    "Mawson-Zeit";
  }
}
"Mexico_Northwest";
{
  "long";
  {
    "generic";
    "Mexiko Nordwestliche Zone-Zeit",
    "standard";
    "Mexiko Nordwestliche Zone-Normalzeit",
    "daylight";
    "Mexiko Nordwestliche Zone-Sommerzeit";
  }
}
"Mexico_Pacific";
```

```
{
  "long";
  {
    "generic";
    "Mexiko Pazifikzone-Zeit",
    "standard";
    "Mexiko Pazifikzone-Normalzeit",
    "daylight";
    "Mexiko Pazifikzone-Sommerzeit";
  }
}
"Mongolia";
{
  "long";
  {
    "generic";
    "Ulaanbaatar-Zeit",
    "standard";
    "Ulaanbaatar-Normalzeit",
    "daylight";
    "Ulaanbaatar-Sommerzeit";
  }
}
"Moscow";
{
  "long";
  {
    "generic";
    "Moskauer Zeit",
    "standard";
    "Moskauer Normalzeit",
    "daylight";
    "Moskauer Sommerzeit";
  }
}
"Myanmar";
{
  "long";
  {
    "standard";
    "Myanmar-Zeit";
  }
}
"Nauru";
{
  "long";
  {
    "standard";
    "Nauru-Zeit";
  }
}
"Nepal";
{
  "long";
  {
    "standard";
    "Nepalesische Zeit";
  }
}
```

```
    }  
  }  
  "New_Caledonia";  
  {  
    "long";  
    {  
      "generic";  
      "Neukaledonische Zeit",  
        "standard";  
      "Neukaledonische Normalzeit",  
        "daylight";  
      "Neukaledonische Sommerzeit";  
    }  
  }  
  "New_Zealand";  
  {  
    "long";  
    {  
      "generic";  
      "Neuseeland-Zeit",  
        "standard";  
      "Neuseeland-Normalzeit",  
        "daylight";  
      "Neuseeland-Sommerzeit";  
    }  
  }  
  "Newfoundland";  
  {  
    "long";  
    {  
      "generic";  
      "Neufundland-Zeit",  
        "standard";  
      "Neufundland-Normalzeit",  
        "daylight";  
      "Neufundland-Sommerzeit";  
    }  
  }  
  "Niue";  
  {  
    "long";  
    {  
      "standard";  
      "Niue-Zeit";  
    }  
  }  
  "Norfolk";  
  {  
    "long";  
    {  
      "standard";  
      "Norfolkinsel-Zeit";  
    }  
  }  
  "Noronha";  
  {  
    "long";
```



```

        {
            "generic";
            "Fernando de Noronha-Zeit",
            "standard";
            "Fernando de Noronha-Normalzeit",
            "daylight";
            "Fernando de Noronha-Sommerzeit";
        }
    }
    "North_Mariana";
    {
        "long";
        {
            "standard";
            "Nördliche-Marianen-Zeit";
        }
    }
    "Novosibirsk";
    {
        "long";
        {
            "generic";
            "Nowosibirsk-Zeit",
            "standard";
            "Nowosibirsk-Normalzeit",
            "daylight";
            "Nowosibirsk-Sommerzeit";
        }
    }
    "Omsk";
    {
        "long";
        {
            "generic";
            "Omsk-Zeit",
            "standard";
            "Omsk-Normalzeit",
            "daylight";
            "Omsk-Sommerzeit";
        }
    }
    "Pakistan";
    {
        "long";
        {
            "generic";
            "Pakistanische Zeit",
            "standard";
            "Pakistanische Normalzeit",
            "daylight";
            "Pakistanische Sommerzeit";
        }
    }
    "Palau";
    {
        "long";
        {

```

```
        "standard";
        "Palau-Zeit";
    }
}
"Papua_New_Guinea";
{
    "long";
    {
        "standard";
        "Papua-Neuguinea-Zeit";
    }
}
"Paraguay";
{
    "long";
    {
        "generic";
        "Paraguayanische Zeit",
        "standard";
        "Paraguayanische Normalzeit",
        "daylight";
        "Paraguayanische Sommerzeit";
    }
}
"Peru";
{
    "long";
    {
        "generic";
        "Peruanische Zeit",
        "standard";
        "Peruanische Normalzeit",
        "daylight";
        "Peruanische Sommerzeit";
    }
}
"Philippines";
{
    "long";
    {
        "generic";
        "Philippinische Zeit",
        "standard";
        "Philippinische Normalzeit",
        "daylight";
        "Philippinische Sommerzeit";
    }
}
"Phoenix_Islands";
{
    "long";
    {
        "standard";
        "Phoenixinseln-Zeit";
    }
}
"Pierre_Miquelon";
```

```

    {
      "long";
      {
        "generic";
        "Saint-Pierre-und-Miquelon-Zeit",
        "standard";
        "Saint-Pierre-und-Miquelon-Normalzeit",
        "daylight";
        "Saint-Pierre-und-Miquelon-Sommerzeit";
      }
    }
    "Pitcairn";
    {
      "long";
      {
        "standard";
        "Pitcairninsele-Zeit";
      }
    }
    "Ponape";
    {
      "long";
      {
        "standard";
        "Ponape-Zeit";
      }
    }
    "Pyongyang";
    {
      "long";
      {
        "standard";
        "Pjöngjang-Zeit";
      }
    }
    "Qyzylorda";
    {
      "long";
      {
        "generic";
        "Qysylorda-Zeit",
        "standard";
        "Qysylorda-Normalzeit",
        "daylight";
        "Qysylorda-Sommerzeit";
      }
    }
    "Reunion";
    {
      "long";
      {
        "standard";
        "Réunion-Zeit";
      }
    }
    "Rothera";
    {

```

```
        "long";
        {
            "standard";
            "Rothera-Zeit";
        }
    }
    "Sakhalin";
    {
        "long";
        {
            "generic";
            "Sachalin-Zeit",
            "standard";
            "Sachalin-Normalzeit",
            "daylight";
            "Sachalin-Sommerzeit";
        }
    }
    "Samara";
    {
        "long";
        {
            "generic";
            "Samara-Zeit",
            "standard";
            "Samara-Normalzeit",
            "daylight";
            "Samara-Sommerzeit";
        }
    }
    "Samoa";
    {
        "long";
        {
            "generic";
            "Samoa-Zeit",
            "standard";
            "Samoa-Normalzeit",
            "daylight";
            "Samoa-Sommerzeit";
        }
    }
    "Seychelles";
    {
        "long";
        {
            "standard";
            "Seychellen-Zeit";
        }
    }
    "Singapore";
    {
        "long";
        {
            "standard";
            "Singapur-Zeit";
        }
    }
}
```

```
}
"Solomon";
{
  "long";
  {
    "standard";
    "Salomoninseln-Zeit";
  }
}
"South_Georgia";
{
  "long";
  {
    "standard";
    "Südgeorgische Zeit";
  }
}
"Suriname";
{
  "long";
  {
    "standard";
    "Suriname-Zeit";
  }
}
"Syowa";
{
  "long";
  {
    "standard";
    "Syowa-Zeit";
  }
}
"Tahiti";
{
  "long";
  {
    "standard";
    "Tahiti-Zeit";
  }
}
"Taipei";
{
  "long";
  {
    "generic";
    "Taipeh-Zeit",
    "standard";
    "Taipeh-Normalzeit",
    "daylight";
    "Taipeh-Sommerzeit";
  }
}
"Tajikistan";
{
  "long";
  {
```

```
        "standard";
        "Tadschikistan-Zeit";
    }
}
"Tokelau";
{
    "long";
    {
        "standard";
        "Tokelau-Zeit";
    }
}
"Tonga";
{
    "long";
    {
        "generic";
        "Tonganische Zeit",
        "standard";
        "Tonganische Normalzeit",
        "daylight";
        "Tonganische Sommerzeit";
    }
}
"Truk";
{
    "long";
    {
        "standard";
        "Chuuk-Zeit";
    }
}
"Turkmenistan";
{
    "long";
    {
        "generic";
        "Turkmenistan-Zeit",
        "standard";
        "Turkmenistan-Normalzeit",
        "daylight";
        "Turkmenistan-Sommerzeit";
    }
}
"Tuvalu";
{
    "long";
    {
        "standard";
        "Tuvalu-Zeit";
    }
}
"Uruguay";
{
    "long";
    {
        "generic";
```

```

        "Uruguayische Zeit",
        "standard";
        "Uruguayische Normalzeit",
        "daylight";
        "Uruguayische Sommerzeit";
    }
}
"Uzbekistan";
{
    "long";
    {
        "generic";
        "Usbekistan-Zeit",
        "standard";
        "Usbekistan-Normalzeit",
        "daylight";
        "Usbekistan-Sommerzeit";
    }
}
"Vanuatu";
{
    "long";
    {
        "generic";
        "Vanuatu-Zeit",
        "standard";
        "Vanuatu-Normalzeit",
        "daylight";
        "Vanuatu-Sommerzeit";
    }
}
"Venezuela";
{
    "long";
    {
        "standard";
        "Venezuela-Zeit";
    }
}
"Vladivostok";
{
    "long";
    {
        "generic";
        "Wladiwostok-Zeit",
        "standard";
        "Wladiwostok-Normalzeit",
        "daylight";
        "Wladiwostok-Sommerzeit";
    }
}
"Volgograd";
{
    "long";
    {
        "generic";
        "Wolgograd-Zeit",

```

```
        "standard";
        "Wolgograd-Normalzeit",
        "daylight";
        "Wolgograd-Sommerzeit";
    }
}
"Vostok";
{
    "long";
    {
        "standard";
        "Wostok-Zeit";
    }
}
"Wake";
{
    "long";
    {
        "standard";
        "Wake-Insel-Zeit";
    }
}
"Wallis";
{
    "long";
    {
        "standard";
        "Wallis-und-Futuna-Zeit";
    }
}
"Yakutsk";
{
    "long";
    {
        "generic";
        "Jakutsk-Zeit",
        "standard";
        "Jakutsk-Normalzeit",
        "daylight";
        "Jakutsk-Sommerzeit";
    }
}
"Yekaterinburg";
{
    "long";
    {
        "generic";
        "Jekaterinburg-Zeit",
        "standard";
        "Jekaterinburg-Normalzeit",
        "daylight";
        "Jekaterinburg-Sommerzeit";
    }
}
}
}
```



```
}  
}  
}
```

[Functional-component]

CA-GREGORIAN.JSON

```
{  
  "main": {  
    "de": {  
      "identity": {  
        "version": {  
          "_number": "$Revision: 12879 $",  
          "_cldrVersion": "30.0.3"  
        },  
        "language": "de"  
      },  
      "dates": {  
        "calendars": {  
          "gregorian": {  
            "months": {  
              "format": {  
                "abbreviated": {  
                  "1": "Jan.",  
                  "2": "Feb.",  
                  "3": "März",  
                  "4": "Apr.",  
                  "5": "Mai",  
                  "6": "Juni",  
                  "7": "Juli",  
                  "8": "Aug.",  
                  "9": "Sep.",  
                  "10": "Okt.",  
                  "11": "Nov.",  
                  "12": "Dez."  
                },  
                "narrow": {  
                  "1": "J",  
                  "2": "F",  
                  "3": "M",  
                  "4": "A",  
                  "5": "M",  
                  "6": "J",  
                  "7": "J",  
                  "8": "A",  
                  "9": "S",  
                  "10": "O",  
                  "11": "N",  
                  "12": "D"  
                },  
                "wide": {  
                  "1": "Januar",  
                  "2": "Februar",  
                  "3": "März",  
                  "4": "April",  
                  "5": "Mai",  
                  "6": "Juni",  
                  "7": "Juli",  
                  "8": "August",  
                  "9": "September",  
                  "10": "Oktober",  
                  "11": "November",  
                  "12": "Dezember"  
                }  
              }  
            }  
          }  
        }  
      }  
    }  
  }  
}
```

```

        "5": "Mai",
        "6": "Juni",
        "7": "Juli",
        "8": "August",
        "9": "September",
        "10": "Oktober",
        "11": "November",
        "12": "Dezember"
      }
    },
    "stand-alone": {
      "abbreviated": {
        "1": "Jan",
        "2": "Feb",
        "3": "Mär",
        "4": "Apr",
        "5": "Mai",
        "6": "Jun",
        "7": "Jul",
        "8": "Aug",
        "9": "Sep",
        "10": "Okt",
        "11": "Nov",
        "12": "Dez"
      },
      "narrow": {
        "1": "J",
        "2": "F",
        "3": "M",
        "4": "A",
        "5": "M",
        "6": "J",
        "7": "J",
        "8": "A",
        "9": "S",
        "10": "O",
        "11": "N",
        "12": "D"
      },
      "wide": {
        "1": "Januar",
        "2": "Februar",
        "3": "März",
        "4": "April",
        "5": "Mai",
        "6": "Juni",
        "7": "Juli",
        "8": "August",
        "9": "September",
        "10": "Oktober",
        "11": "November",
        "12": "Dezember"
      }
    }
  },
  "days": {
    "format": {

```

```
    "abbreviated": {
      "sun": "So.",
      "mon": "Mo.",
      "tue": "Di.",
      "wed": "Mi.",
      "thu": "Do.",
      "fri": "Fr.",
      "sat": "Sa."
    },
    "narrow": {
      "sun": "S",
      "mon": "M",
      "tue": "D",
      "wed": "M",
      "thu": "D",
      "fri": "F",
      "sat": "S"
    },
    "short": {
      "sun": "So.",
      "mon": "Mo.",
      "tue": "Di.",
      "wed": "Mi.",
      "thu": "Do.",
      "fri": "Fr.",
      "sat": "Sa."
    },
    "wide": {
      "sun": "Sonntag",
      "mon": "Montag",
      "tue": "Dienstag",
      "wed": "Mittwoch",
      "thu": "Donnerstag",
      "fri": "Freitag",
      "sat": "Samstag"
    }
  },
  "stand-alone": {
    "abbreviated": {
      "sun": "So",
      "mon": "Mo",
      "tue": "Di",
      "wed": "Mi",
      "thu": "Do",
      "fri": "Fr",
      "sat": "Sa"
    },
    "narrow": {
      "sun": "S",
      "mon": "M",
      "tue": "D",
      "wed": "M",
      "thu": "D",
      "fri": "F",
      "sat": "S"
    },
    "short": {
```

```

        "sun": "So.",
        "mon": "Mo.",
        "tue": "Di.",
        "wed": "Mi.",
        "thu": "Do.",
        "fri": "Fr.",
        "sat": "Sa."
    },
    "wide": {
        "sun": "Sonntag",
        "mon": "Montag",
        "tue": "Dienstag",
        "wed": "Mittwoch",
        "thu": "Donnerstag",
        "fri": "Freitag",
        "sat": "Samstag"
    }
},
"quarters": {
    "format": {
        "abbreviated": {
            "1": "Q1",
            "2": "Q2",
            "3": "Q3",
            "4": "Q4"
        },
        "narrow": {
            "1": "1",
            "2": "2",
            "3": "3",
            "4": "4"
        },
        "wide": {
            "1": "1. Quartal",
            "2": "2. Quartal",
            "3": "3. Quartal",
            "4": "4. Quartal"
        }
    },
    "stand-alone": {
        "abbreviated": {
            "1": "Q1",
            "2": "Q2",
            "3": "Q3",
            "4": "Q4"
        },
        "narrow": {
            "1": "1",
            "2": "2",
            "3": "3",
            "4": "4"
        },
        "wide": {
            "1": "1. Quartal",
            "2": "2. Quartal",
            "3": "3. Quartal",

```

```

        "4": "4. Quartal"
      }
    }
  },
  "dayPeriods": {
    "format": {
      "abbreviated": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "morgens",
        "morning2": "vormittags",
        "afternoon1": "mittags",
        "afternoon2": "nachmittags",
        "evening1": "abends",
        "night1": "nachts"
      },
      "narrow": {
        "midnight": "Mitternacht",
        "am": "vm.",
        "pm": "nm.",
        "morning1": "morgens",
        "morning2": "vormittags",
        "afternoon1": "mittags",
        "afternoon2": "nachmittags",
        "evening1": "abends",
        "night1": "nachts"
      },
      "wide": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "morgens",
        "morning2": "vormittags",
        "afternoon1": "mittags",
        "afternoon2": "nachmittags",
        "evening1": "abends",
        "night1": "nachts"
      }
    },
    "stand-alone": {
      "abbreviated": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "Morgen",
        "morning2": "Vormittag",
        "afternoon1": "Mittag",
        "afternoon2": "Nachmittag",
        "evening1": "Abend",
        "night1": "Nacht"
      },
      "narrow": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "Morgen",

```

```

        "morning2": "Vormittag",
        "afternoon1": "Mittag",
        "afternoon2": "Nachmittag",
        "evening1": "Abend",
        "night1": "Nacht"
    },
    "wide": {
        "midnight": "Mitternacht",
        "am": "vorm.",
        "pm": "nachm.",
        "morning1": "Morgen",
        "morning2": "Vormittag",
        "afternoon1": "Mittag",
        "afternoon2": "Nachmittag",
        "evening1": "Abend",
        "night1": "Nacht"
    }
},
"eras": {
    "eraNames": {
        "0": "v. Chr.",
        "0-alt-variant": "vor unserer Zeitrechnung",
        "1": "n. Chr.",
        "1-alt-variant": "unserer Zeitrechnung"
    },
    "eraAbbr": {
        "0": "v. Chr.",
        "0-alt-variant": "v. u. Z.",
        "1": "n. Chr.",
        "1-alt-variant": "u. Z."
    },
    "eraNarrow": {
        "0": "v. Chr.",
        "0-alt-variant": "v. u. Z.",
        "1": "n. Chr.",
        "1-alt-variant": "u. Z."
    }
},
"dateFormats": {
    "full": "EEEE, d. MMMM y",
    "long": "d. MMMM y",
    "medium": "dd.MM.y",
    "short": "dd.MM.yy"
},
"timeFormats": {
    "full": "HH:mm:ss zzzz",
    "long": "HH:mm:ss z",
    "medium": "HH:mm:ss",
    "short": "HH:mm"
},
"dateTimeFormats": {
    "full": "{1} 'um' {0}",
    "long": "{1} 'um' {0}",
    "medium": "{1}, {0}",
    "short": "{1}, {0}",
    "availableFormats": {

```

```

    "d": "d",
    "E": "ccc",
    "Ed": "E, d.",
    "Ehm": "E h:mm a",
    "EHm": "E, HH:mm",
    "Ehms": "E, h:mm:ss a",
    "EHms": "E, HH:mm:ss",
    "Gy": "y G",
    "GyMMM": "MMM y G",
    "GyMMMd": "d. MMM y G",
    "GyMMMED": "E, d. MMM y G",
    "h": "h 'Uhr' a",
    "H": "HH 'Uhr'",
    "hm": "h:mm a",
    "Hm": "HH:mm",
    "hms": "h:mm:ss a",
    "Hms": "HH:mm:ss",
    "hmsv": "h:mm:ss a v",
    "Hmsv": "HH:mm:ss v",
    "hmv": "h:mm a v",
    "Hmv": "HH:mm v",
    "M": "L",
    "Md": "d.M.",
    "MED": "E, d.M.",
    "MMd": "d.MM.",
    "MMdd": "dd.MM.",
    "MMM": "LLL",
    "MMMd": "d. MMM",
    "MMMED": "E, d. MMM",
    "MMMMd": "d. MMMM",
    "MMMMMED": "E, d. MMMM",
    "MMMMMW": "'Woche' W 'im' MMM",
    "MMMMMW": "'Woche' W 'im' MMM",
    "ms": "mm:ss",
    "y": "y",
    "yM": "M.y",
    "yMd": "d.M.y",
    "yMED": "E, d.M.y",
    "yMM": "MM.y",
    "yMMdd": "dd.MM.y",
    "yMMM": "MMM y",
    "yMMMd": "d. MMM y",
    "yMMMED": "E, d. MMM y",
    "yMMMM": "MMMM y",
    "yQQQ": "QQQ y",
    "yQQQQ": "QQQQ y",
    "yw": "'Woche' w 'des' 'Jahres' y",
    "yw": "'Woche' w 'des' 'Jahres' y"
  },
  "appendItems": {
    "Day": "{0} ({2}: {1})",
    "Day-Of-Week": "{0} {1}",
    "Era": "{1} {0}",
    "Hour": "{0} ({2}: {1})",
    "Minute": "{0} ({2}: {1})",
    "Month": "{0} ({2}: {1})",
    "Quarter": "{0} ({2}: {1})",
  }

```

```

    "Second": "{0} ({2}: {1})",
    "Timezone": "{0} {1}",
    "Week": "{0} ({2}: {1})",
    "Year": "{1} {0}"
  },
  "intervalFormats": {
    "intervalFormatFallback": "{0} - {1}",
    "d": {
      "d": "d.-d."
    },
    "h": {
      "a": "h 'Uhr' a - h 'Uhr' a",
      "h": "h - h 'Uhr' a"
    },
    "H": {
      "H": "HH-HH 'Uhr'"
    },
    "hm": {
      "a": "h:mm a - h:mm a",
      "h": "h:mm-h:mm a",
      "m": "h:mm-h:mm a"
    },
    "Hm": {
      "H": "HH:mm-HH:mm 'Uhr'",
      "m": "HH:mm-HH:mm 'Uhr'"
    },
    "hmv": {
      "a": "h:mm a - h:mm a v",
      "h": "h:mm-h:mm a v",
      "m": "h:mm-h:mm a v"
    },
    "Hmv": {
      "H": "HH:mm-HH:mm 'Uhr' v",
      "m": "HH:mm-HH:mm 'Uhr' v"
    },
    "hv": {
      "a": "h a - h a v",
      "h": "h-h a v"
    },
    "Hv": {
      "H": "HH-HH 'Uhr' v"
    },
    "M": {
      "M": "M.-M."
    },
    "Md": {
      "d": "dd.MM. - dd.MM.",
      "M": "dd.MM. - dd.MM."
    },
    "MEd": {
      "d": "E, dd.MM. - E, dd.MM.",
      "M": "E, dd.MM. - E, dd.MM."
    },
    "MMM": {
      "M": "MMM-MMM"
    },
    "MMMd": {

```



```
"d": "d.-d. MMM",  
"M": "d. MMM - d. MMM"  
},  
"MMMEd": {  
    "d": "E, d. - E, d. MMM",  
    "M": "E, d. MMM - E, d. MMM"  
},  
"MMMM": {  
    "M": "LLLL-LLLL"  
},  
"Y": {  
    "Y": "Y-Y"  
},  
"YM": {  
    "M": "MM.y - MM.y",  
    "y": "MM.y - MM.y"  
},  
"yMd": {  
    "d": "dd.MM.y - dd.MM.y",  
    "M": "dd.MM.y - dd.MM.y",  
    "y": "dd.MM.y - dd.MM.y"  
},  
"yMED": {  
    "d": "E, dd.MM.y - E, dd.MM.y",  
    "M": "E, dd.MM.y - E, dd.MM.y",  
    "y": "E, dd.MM.y - E, dd.MM.y"  
},  
"yMMM": {  
    "M": "MMM-MMM y",  
    "y": "MMM y - MMM y"  
},  
"yMMMd": {  
    "d": "d.-d. MMM y",  
    "M": "d. MMM - d. MMM y",  
    "y": "d. MMM y - d. MMM y"  
},  
"yMMMEd": {  
    "d": "E, d. - E, d. MMM y",  
    "M": "E, d. MMM - E, d. MMM y",  
    "y": "E, d. MMM y - E, d. MMM y"  
},  
"yMMMM": {  
    "M": "MMMM-MMMM y",  
    "y": "MMMM y - MMMM y"  
}  
  
}  
  
}  
  
}  
  
}
```

CA-GREGORIAN.JSX

```

{
  "main";
  {
    "de";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 12879 $",
          "_cldrVersion";
          "30.0.3";
        }
        "language";
        "de";
      }
      "dates";
      {
        "calendars";
        {
          "gregorian";
          {
            "months";
            {
              "format";
              {
                "abbreviated";
                {
                  "1";
                  "Jan.",
                  "2";
                  "Feb.",
                  "3";
                  "März",
                  "4";
                  "Apr.",
                  "5";
                  "Mai",
                  "6";
                  "Juni",
                  "7";
                  "Juli",
                  "8";
                  "Aug.",
                  "9";
                  "Sep.",
                  "10";
                  "Okt.",
                  "11";
                  "Nov.",
                  "12";
                  "Dez.";
                }
                "narrow";
                {
                  "1";

```

```
        "J", "2";
        "F", "3";
        "M", "4";
        "A", "5";
        "M", "6";
        "J", "7";
        "J", "8";
        "A", "9";
        "S", "10";
        "O", "11";
        "N", "12";
        "D";
    }
    "wide";
    {
        "1";
        "Januar",
        "2";
        "Februar",
        "3";
        "März",
        "4";
        "April",
        "5";
        "Mai",
        "6";
        "Juni",
        "7";
        "Juli",
        "8";
        "August",
        "9";
        "September",
        "10";
        "Oktober",
        "11";
        "November",
        "12";
        "Dezember";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
```

```
        "Jan",
        "2";
        "Feb",
        "3";
        "Mär",
        "4";
        "Apr",
        "5";
        "Mai",
        "6";
        "Jun",
        "7";
        "Jul",
        "8";
        "Aug",
        "9";
        "Sep",
        "10";
        "Okt",
        "11";
        "Nov",
        "12";
        "Dez";
    }
    "narrow";
    {
        "1";
        "J",
        "2";
        "F",
        "3";
        "M",
        "4";
        "A",
        "5";
        "M",
        "6";
        "J",
        "7";
        "J",
        "8";
        "A",
        "9";
        "S",
        "10";
        "O",
        "11";
        "N",
        "12";
        "D";
    }
    "wide";
    {
        "1";
        "Januar",
        "2";
        "Februar",
```

```
        "3";
        "März",
        "4";
        "April",
        "5";
        "Mai",
        "6";
        "Juni",
        "7";
        "Juli",
        "8";
        "August",
        "9";
        "September",
        "10";
        "Oktober",
        "11";
        "November",
        "12";
        "Dezember";
    }
}
}
"days";
{
    "format";
    {
        "abbreviated";
        {
            "sun";
            "So.",
            "mon";
            "Mo.",
            "tue";
            "Di.",
            "wed";
            "Mi.",
            "thu";
            "Do.",
            "fri";
            "Fr.",
            "sat";
            "Sa.";
        }
        "narrow";
        {
            "sun";
            "S",
            "mon";
            "M",
            "tue";
            "D",
            "wed";
            "M",
            "thu";
            "D",
            "fri";
```

```
        "F",
        "sat";
        "S";
    }
    "short";
    {
        "sun";
        "So.",
        "mon";
        "Mo.",
        "tue";
        "Di.",
        "wed";
        "Mi.",
        "thu";
        "Do.",
        "fri";
        "Fr.",
        "sat";
        "Sa.";
    }
    "wide";
    {
        "sun";
        "Sonntag",
        "mon";
        "Montag",
        "tue";
        "Dienstag",
        "wed";
        "Mittwoch",
        "thu";
        "Donnerstag",
        "fri";
        "Freitag",
        "sat";
        "Samstag";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "sun";
        "So",
        "mon";
        "Mo",
        "tue";
        "Di",
        "wed";
        "Mi",
        "thu";
        "Do",
        "fri";
        "Fr",
        "sat";
        "Sa";
    }
}
```

```
}
"narrow";
{
  "sun";
  "S", "mon";
  "M", "tue";
  "D", "wed";
  "M", "thu";
  "D", "fri";
  "F", "sat";
  "S";
}
"short";
{
  "sun";
  "So.", "mon";
  "Mo.", "tue";
  "Di.", "wed";
  "Mi.", "thu";
  "Do.", "fri";
  "Fr.", "sat";
  "Sa.";
}
"wide";
{
  "sun";
  "Sonntag", "mon";
  "Montag", "tue";
  "Dienstag", "wed";
  "Mittwoch", "thu";
  "Donnerstag", "fri";
  "Freitag", "sat";
  "Samstag";
}
}
"quarters";
{
  "format";
```

```
{
  "abbreviated":
  {
    "1";
    "Q1",
    "2";
    "Q2",
    "3";
    "Q3",
    "4";
    "Q4";
  }
  "narrow":
  {
    "1";
    "1",
    "2";
    "2",
    "3";
    "3",
    "4";
    "4";
  }
  "wide":
  {
    "1";
    "1. Quartal",
    "2";
    "2. Quartal",
    "3";
    "3. Quartal",
    "4";
    "4. Quartal";
  }
}
"stand-alone":
{
  "abbreviated":
  {
    "1";
    "Q1",
    "2";
    "Q2",
    "3";
    "Q3",
    "4";
    "Q4";
  }
  "narrow":
  {
    "1";
    "1",
    "2";
    "2",
    "3";
    "3",
    "4";
  }
}
```



```
        "4";
    }
    "wide";
    {
        "1";
        "1. Quartal",
        "2";
        "2. Quartal",
        "3";
        "3. Quartal",
        "4";
        "4. Quartal";
    }
}
"dayPeriods";
{
    "format";
    {
        "abbreviated";
        {
            "midnight";
            "Mitternacht",
            "am";
            "vorm.",
            "pm";
            "nachm.",
            "morning1";
            "morgens",
            "morning2";
            "vormittags",
            "afternoon1";
            "mittags",
            "afternoon2";
            "nachmittags",
            "evening1";
            "abends",
            "night1";
            "nachts";
        }
    }
    "narrow";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vm.",
        "pm";
        "nm.",
        "morning1";
        "morgens",
        "morning2";
        "vormittags",
        "afternoon1";
        "mittags",
        "afternoon2";
        "nachmittags",
        "evening1";
    }
}
```

```
        "abends",
        "night1";
        "nachts";
    }
    "wide";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vorm.",
        "pm";
        "nachm.",
        "morning1";
        "morgens",
        "morning2";
        "vormittags",
        "afternoon1";
        "mittags",
        "afternoon2";
        "nachmittags",
        "evening1";
        "abends",
        "night1";
        "nachts";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vorm.",
        "pm";
        "nachm.",
        "morning1";
        "Morgen",
        "morning2";
        "Vormittag",
        "afternoon1";
        "Mittag",
        "afternoon2";
        "Nachmittag",
        "evening1";
        "Abend",
        "night1";
        "Nacht";
    }
    "narrow";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vorm.",
        "pm";
        "nachm.",
```

```

        "morning1";
        "Morgen",
        "morning2";
        "Vormittag",
        "afternoon1";
        "Mittag",
        "afternoon2";
        "Nachmittag",
        "evening1";
        "Abend",
        "night1";
        "Nacht";
    }
    "wide";
    {
        "midnight";
        "Mitternacht",
        "am";
        "vorm.",
        "pm";
        "nachm.",
        "morning1";
        "Morgen",
        "morning2";
        "Vormittag",
        "afternoon1";
        "Mittag",
        "afternoon2";
        "Nachmittag",
        "evening1";
        "Abend",
        "night1";
        "Nacht";
    }
}
"eras";
{
    "eraNames";
    {
        "0";
        "v. Chr.",
        "0-alt-variant";
        "vor unserer Zeitrechnung",
        "1";
        "n. Chr.",
        "1-alt-variant";
        "unserer Zeitrechnung";
    }
    "eraAbbr";
    {
        "0";
        "v. Chr.",
        "0-alt-variant";
        "v. u. Z.",
        "1";
        "n. Chr.",
    }
}

```

```

        "1-alt-variant";
        "u. Z.";
    }
    "eraNarrow";
    {
        "0";
        "v. Chr.",
        "0-alt-variant";
        "v. u. Z.",
        "1";
        "n. Chr.",
        "1-alt-variant";
        "u. Z.";
    }
}
"dateFormats";
{
    "full";
    "EEEE, d. MMMM y",
    "long";
    "d. MMMM y",
    "medium";
    "dd.MM.y",
    "short";
    "dd.MM.yy";
}
"timeFormats";
{
    "full";
    "HH:mm:ss zzzz",
    "long";
    "HH:mm:ss z",
    "medium";
    "HH:mm:ss",
    "short";
    "HH:mm";
}
"dateTimeFormats";
{
    "full";
    "{1} 'um' {0}",
    "long";
    "{1} 'um' {0}",
    "medium";
    "{1}, {0}",
    "short";
    "{1}, {0}",
    "availableFormats";
    {
        "d";
        "d",
        "E";
        "ccc",
        "Ed";
        "E, d.",
        "Ehm";
        "E h:mm a",
    }
}

```

```

        "EHm";
    "E, HH:mm",
        "Ehms";
    "E, h:mm:ss a",
        "EHms";
    "E, HH:mm:ss",
        "Gy";
    "y G",
        "GyMMM";
    "MMM y G",
        "GyMMMd";
    "d. MMM y G",
        "GyMMMED";
    "E, d. MMM y G",
        "h";
    "h 'Uhr' a",
        "H";
    "HH 'Uhr' ",
        "hm";
    "h:mm a",
        "Hm";
    "HH:mm",
        "hms";
    "h:mm:ss a",
        "Hms";
    "HH:mm:ss",
        "hmsv";
    "h:mm:ss a v",
        "Hmsv";
    "HH:mm:ss v",
        "hmv";
    "h:mm a v",
        "Hmv";
    "HH:mm v",
        "M";
    "L",
        "Md";
    "d.M.",
        "MED";
    "E, d.M.",
        "MMd";
    "d.MM.",
        "MMdd";
    "dd.MM.",
        "MMM";
    "LLL",
        "MMMd";
    "d. MMM",
        "MMMED";
    "E, d. MMM",
        "MMMMd";
    "d. MMMM",
        "MMMMED";
    "E, d. MMMM",
        "MMMMW";
    "'Woche' W 'im' MMM",
        "MMMMW";

```

```

        "'Woche' W 'im' MMM",
        "ms";
        "mm:ss",
        "y";
        "Y",
        "yM";
        "M.y",
        "yMd";
        "d.M.y",
        "yMEd";
        "E, d.M.y",
        "yMM";
        "MM.y",
        "yMMdd";
        "dd.MM.y",
        "yMMM";
        "MMM y",
        "yMMMd";
        "d. MMM y",
        "yMMMEd";
        "E, d. MMM y",
        "yMMMM";
        "MMMM y",
        "yQQQ";
        "QQQ y",
        "yQQQQ";
        "QQQQ y",
        "yw";
        "'Woche' w 'des' 'Jahres' y",
        "yw";
        "'Woche' w 'des' 'Jahres' y";
    }
    "appendItems";
    {
        "Day";
        "{0} ({2}: {1})",
        "Day-Of-Week";
        "{0} {1}",
        "Era";
        "{1} {0}",
        "Hour";
        "{0} ({2}: {1})",
        "Minute";
        "{0} ({2}: {1})",
        "Month";
        "{0} ({2}: {1})",
        "Quarter";
        "{0} ({2}: {1})",
        "Second";
        "{0} ({2}: {1})",
        "Timezone";
        "{0} {1}",
        "Week";
        "{0} ({2}: {1})",
        "Year";
        "{1} {0}";
    }
}

```

```

"intervalFormats";
{
  "intervalFormatFallback";
  "{0} - {1}",
  "d";
  {
    "d";
    "d.-d.";
  }
  "h";
  {
    "a";
    "h 'Uhr' a - h 'Uhr' a",
    "h";
    "h - h 'Uhr' a";
  }
  "H";
  {
    "H";
    "HH-HH 'Uhr'";
  }
  "hm";
  {
    "a";
    "h:mm a - h:mm a",
    "h";
    "h:mm-h:mm a",
    "m";
    "h:mm-h:mm a";
  }
  "Hm";
  {
    "H";
    "HH:mm-HH:mm 'Uhr'",
    "m";
    "HH:mm-HH:mm 'Uhr'";
  }
  "hmv";
  {
    "a";
    "h:mm a - h:mm a v",
    "h";
    "h:mm-h:mm a v",
    "m";
    "h:mm-h:mm a v";
  }
  "Hmv";
  {
    "H";
    "HH:mm-HH:mm 'Uhr' v",
    "m";
    "HH:mm-HH:mm 'Uhr' v";
  }
  "hv";
  {
    "a";
    "h a - h a v",

```

```

        "h";
        "h-h a v";
    }
    "Hv";
    {
        "H";
        "HH-HH 'Uhr' v";
    }
    "M";
    {
        "M";
        "M.-M.";
    }
    "Md";
    {
        "d";
        "dd.MM. - dd.MM.",
        "M";
        "dd.MM. - dd.MM.";
    }
    "MEd";
    {
        "d";
        "E, dd.MM. - E, dd.MM.",
        "M";
        "E, dd.MM. - E, dd.MM.";
    }
    "MMM";
    {
        "M";
        "MMM-MMM";
    }
    "MMMd";
    {
        "d";
        "d.-d. MMM",
        "M";
        "d. MMM - d. MMM";
    }
    "MMMEd";
    {
        "d";
        "E, d. - E, d. MMM",
        "M";
        "E, d. MMM - E, d. MMM";
    }
    "MMMM";
    {
        "M";
        "LLLL-LLLL";
    }
    "Y";
    {
        "Y";
        "Y-Y";
    }
    "YM";

```



```

    {
        "M";
        "MM.y - MM.y",
        "Y";
        "MM.y - MM.y";
    }
    "yMd";
    {
        "d";
        "dd.MM.y - dd.MM.y",
        "M";
        "dd.MM.y - dd.MM.y",
        "Y";
        "dd.MM.y - dd.MM.y";
    }
    "yMEd";
    {
        "d";
        "E, dd.MM.y - E, dd.MM.y",
        "M";
        "E, dd.MM.y - E, dd.MM.y",
        "Y";
        "E, dd.MM.y - E, dd.MM.y";
    }
    "yMMM";
    {
        "M";
        "MMM-MMM y",
        "Y";
        "MMM y - MMM y";
    }
    "yMMMd";
    {
        "d";
        "d.-d. MMM y",
        "M";
        "d. MMM - d. MMM y",
        "Y";
        "d. MMM y - d. MMM y";
    }
    "yMMMEd";
    {
        "d";
        "E, d. - E, d. MMM y",
        "M";
        "E, d. MMM - E, d. MMM y",
        "Y";
        "E, d. MMM y - E, d. MMM y";
    }
    "yMMMM";
    {
        "M";
        "MMMM-MMMM y",
        "Y";
        "MMMM y - MMMM y";
    }
}

```

$$\left\{ \begin{array}{l} \\ \\ \\ \\ \\ \\ \\ \end{array} \right\}$$

CURRENCIES.JSON

```
{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31"
        },
        "language": "de"
      },
      "numbers": {
        "currencies": {
          "ADP": {
            "displayName": "Andorranische Pesete",
            "displayName-count-one": "Andorranische Pesete",
            "displayName-count-other": "Andorranische Peseten",
            "symbol": "ADP"
          },
          "AED": {
            "displayName": "VAE-Dirham",
            "displayName-count-one": "VAE-Dirham",
            "displayName-count-other": "VAE-Dirham",
            "symbol": "AED"
          },
          "AFA": {
            "displayName": "Afghanische Afghani (1927-2002)",
            "displayName-count-one": "Afghanische Afghani (1927-2002)",
            "displayName-count-other": "Afghanische Afghani (1927-2002)",
            "symbol": "AFA"
          },
          "AFN": {
            "displayName": "Afghanischer Afghani",
            "displayName-count-one": "Afghanischer Afghani",
            "displayName-count-other": "Afghanische Afghani",
            "symbol": "AFN"
          },
          "ALK": {
            "displayName": "Albanischer Lek (1946-1965)",
            "displayName-count-one": "Albanischer Lek (1946-1965)",
            "displayName-count-other": "Albanische Lek (1946-1965)"
          },
          "ALL": {
            "displayName": "Albanischer Lek",
            "displayName-count-one": "Albanischer Lek",
            "displayName-count-other": "Albanische Lek",
            "symbol": "ALL"
          }
        }
      }
    }
  }
}
```

```

    },
    "AMD": {
      "displayName": "Armenischer Dram",
      "displayName-count-one": "Armenischer Dram",
      "displayName-count-other": "Armenische Dram",
      "symbol": "AMD"
    },
    "ANG": {
      "displayName": "Niederländische-Antillen-Gulden",
      "displayName-count-one": "Niederländische-Antillen-Gulden",
      "displayName-count-other": "Niederländische-Antillen-Gulden",
      "symbol": "ANG"
    },
    "AOA": {
      "displayName": "Angolanischer Kwanza",
      "displayName-count-one": "Angolanischer Kwanza",
      "displayName-count-other": "Angolanische Kwanza",
      "symbol": "AOA",
      "symbol-alt-narrow": "Kz"
    },
    "AOK": {
      "displayName": "Angolanischer Kwanza (1977-1990)",
      "displayName-count-one": "Angolanischer Kwanza (1977-1990)",
      "displayName-count-other": "Angolanische Kwanza (1977-1990)",
      "symbol": "AOK"
    },
    "AON": {
      "displayName": "Angolanischer Neuer Kwanza (1990-2000)",
      "displayName-count-one": "Angolanischer Neuer Kwanza (1990-2000)",
      "displayName-count-other": "Angolanische Neue Kwanza (1990-2000)",
      "symbol": "AON"
    },
    "AOR": {
      "displayName": "Angolanischer Kwanza Reajustado (1995-1999)",
      "displayName-count-one": "Angolanischer Kwanza Reajustado (1995-1999)",
      "displayName-count-other": "Angolanische Kwanza Reajustado (1995-1999)",
      "symbol": "AOR"
    },
    "ARA": {
      "displayName": "Argentinischer Austral",
      "displayName-count-one": "Argentinischer Austral",
      "displayName-count-other": "Argentinische Austral",
      "symbol": "ARA"
    },
    "ARL": {
      "displayName": "Argentinischer Peso Ley (1970-1983)",
      "displayName-count-one": "Argentinischer Peso Ley (1970-1983)",
      "displayName-count-other": "Argentinische Pesos Ley (1970-1983)",
      "symbol": "ARL"
    },
    "ARM": {
      "displayName": "Argentinischer Peso (1881-1970)",
      "displayName-count-one": "Argentinischer Peso (1881-1970)",

```

```

        "displayName-count-other": "Argentinische Pesos (1881-1970)",
        "symbol": "ARM"
    },
    "ARP": {
        "displayName": "Argentinischer Peso (1983-1985)",
        "displayName-count-one": "Argentinischer Peso (1983-1985)",
        "displayName-count-other": "Argentinische Peso (1983-1985)",
        "symbol": "ARP"
    },
    "ARS": {
        "displayName": "Argentinischer Peso",
        "displayName-count-one": "Argentinischer Peso",
        "displayName-count-other": "Argentinische Pesos",
        "symbol": "ARS",
        "symbol-alt-narrow": "$"
    },
    "ATS": {
        "displayName": "Österreichischer Schilling",
        "displayName-count-one": "Österreichischer Schilling",
        "displayName-count-other": "Österreichische Schilling",
        "symbol": "öS"
    },
    "AUD": {
        "displayName": "Australischer Dollar",
        "displayName-count-one": "Australischer Dollar",
        "displayName-count-other": "Australische Dollar",
        "symbol": "AU$",
        "symbol-alt-narrow": "$"
    },
    "AWG": {
        "displayName": "Aruba-Florin",
        "displayName-count-one": "Aruba-Florin",
        "displayName-count-other": "Aruba-Florin",
        "symbol": "AWG"
    },
    "AZM": {
        "displayName": "Aserbaidtschan-Manat (1993-2006)",
        "displayName-count-one": "Aserbaidtschan-Manat (1993-2006)",
        "displayName-count-other": "Aserbaidtschan-Manat (1993-2006)",
        "symbol": "AZM"
    },
    "AZN": {
        "displayName": "Aserbaidtschan-Manat",
        "displayName-count-one": "Aserbaidtschan-Manat",
        "displayName-count-other": "Aserbaidtschan-Manat",
        "symbol": "AZN"
    },
    "BAD": {
        "displayName": "Bosnien und Herzegowina Dinar (1992-1994)",
        "displayName-count-one": "Bosnien und Herzegowina Dinar (1992-1994)",
        "displayName-count-other": "Bosnien und Herzegowina Dinar (1992-1994)",
        "symbol": "BAD"
    },
    "BAM": {
        "displayName": "Bosnien und Herzegowina Konvertierbare Mark",

```

```

        "displayName-count-one": "Bosnien und Herzegowina Konvertierbare
Mark",
        "displayName-count-other": "Bosnien und Herzegowina
Konvertierbare Mark",
        "symbol": "BAM",
        "symbol-alt-narrow": "KM"
    },
    "BAN": {
        "displayName": "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
        "displayName-count-one": "Bosnien und Herzegowina Neuer Dinar
(1994-1997)",
        "displayName-count-other": "Bosnien und Herzegowina Neue Dinar
(1994-1997)",
        "symbol": "BAN"
    },
    "BBD": {
        "displayName": "Barbados-Dollar",
        "displayName-count-one": "Barbados-Dollar",
        "displayName-count-other": "Barbados-Dollar",
        "symbol": "BBD",
        "symbol-alt-narrow": "$"
    },
    "BDT": {
        "displayName": "Bangladesch-Taka",
        "displayName-count-one": "Bangladesch-Taka",
        "displayName-count-other": "Bangladesch-Taka",
        "symbol": "BDT",
        "symbol-alt-narrow": "৳"
    },
    "BEC": {
        "displayName": "Belgischer Franc (konvertibel)",
        "displayName-count-one": "Belgischer Franc (konvertibel)",
        "displayName-count-other": "Belgische Franc (konvertibel)",
        "symbol": "BEC"
    },
    "BEF": {
        "displayName": "Belgischer Franc",
        "displayName-count-one": "Belgischer Franc",
        "displayName-count-other": "Belgische Franc",
        "symbol": "BEF"
    },
    "BEL": {
        "displayName": "Belgischer Finanz-Franc",
        "displayName-count-one": "Belgischer Finanz-Franc",
        "displayName-count-other": "Belgische Finanz-Franc",
        "symbol": "BEL"
    },
    "BGL": {
        "displayName": "Bulgarische Lew (1962-1999)",
        "displayName-count-one": "Bulgarische Lew (1962-1999)",
        "displayName-count-other": "Bulgarische Lew (1962-1999)",
        "symbol": "BGL"
    },
    "BGM": {
        "displayName": "Bulgarischer Lew (1952-1962)",
        "displayName-count-one": "Bulgarischer Lew (1952-1962)",

```

```

        "displayName-count-other": "Bulgarische Lew (1952-1962)",
        "symbol": "BGK"
    },
    "BGN": {
        "displayName": "Bulgarischer Lew",
        "displayName-count-one": "Bulgarischer Lew",
        "displayName-count-other": "Bulgarische Lew",
        "symbol": "BGN"
    },
    "BGO": {
        "displayName": "Bulgarischer Lew (1879-1952)",
        "displayName-count-one": "Bulgarischer Lew (1879-1952)",
        "displayName-count-other": "Bulgarische Lew (1879-1952)",
        "symbol": "BGJ"
    },
    "BHD": {
        "displayName": "Bahrain-Dinar",
        "displayName-count-one": "Bahrain-Dinar",
        "displayName-count-other": "Bahrain-Dinar",
        "symbol": "BHD"
    },
    "BIF": {
        "displayName": "Burundi-Franc",
        "displayName-count-one": "Burundi-Franc",
        "displayName-count-other": "Burundi-Francs",
        "symbol": "BIF"
    },
    "BMD": {
        "displayName": "Bermuda-Dollar",
        "displayName-count-one": "Bermuda-Dollar",
        "displayName-count-other": "Bermuda-Dollar",
        "symbol": "BMD",
        "symbol-alt-narrow": "$"
    },
    "BND": {
        "displayName": "Brunei-Dollar",
        "displayName-count-one": "Brunei-Dollar",
        "displayName-count-other": "Brunei-Dollar",
        "symbol": "BND",
        "symbol-alt-narrow": "$"
    },
    "BOB": {
        "displayName": "Bolivanischer Boliviano",
        "displayName-count-one": "Bolivanischer Boliviano",
        "displayName-count-other": "Bolivianische Bolivianos",
        "symbol": "BOB",
        "symbol-alt-narrow": "Bs"
    },
    "BOL": {
        "displayName": "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-one": "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-other": "Bolivianische Bolivianos (1863-
1963)",
        "symbol": "BOL"
    },
    "BOP": {
        "displayName": "Bolivianischer Peso",

```

```

        "displayName-count-one": "Bolivianischer Peso",
        "displayName-count-other": "Bolivianische Peso",
        "symbol": "BOP"
    },
    "BOV": {
        "displayName": "Boliviansiche Mvdol",
        "displayName-count-one": "Boliviansiche Mvdol",
        "displayName-count-other": "Bolivianische Mvdol",
        "symbol": "BOV"
    },
    "BRB": {
        "displayName": "Brasilianischer Cruzeiro Novo (1967-1986)",
        "displayName-count-one": "Brasilianischer Cruzeiro Novo (1967-
1986)",
        "displayName-count-other": "Brasilianische Cruzeiro Novo (1967-
1986)",
        "symbol": "BRB"
    },
    "BRC": {
        "displayName": "Brasilianischer Cruzado (1986-1989)",
        "displayName-count-one": "Brasilianischer Cruzado (1986-1989)",
        "displayName-count-other": "Brasilianische Cruzado (1986-1989)",
        "symbol": "BRC"
    },
    "BRE": {
        "displayName": "Brasilianischer Cruzeiro (1990-1993)",
        "displayName-count-one": "Brasilianischer Cruzeiro (1990-1993)",
        "displayName-count-other": "Brasilianische Cruzeiro (1990-1993)",
        "symbol": "BRE"
    },
    "BRL": {
        "displayName": "Brasilianischer Real",
        "displayName-count-one": "Brasilianischer Real",
        "displayName-count-other": "Brasilianische Real",
        "symbol": "R$",
        "symbol-alt-narrow": "R$"
    },
    "BRN": {
        "displayName": "Brasilianischer Cruzado Novo (1989-1990)",
        "displayName-count-one": "Brasilianischer Cruzado Novo (1989-
1990)",
        "displayName-count-other": "Brasilianische Cruzado Novo (1989-
1990)",
        "symbol": "BRN"
    },
    "BRR": {
        "displayName": "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-one": "Brasilianischer Cruzeiro (1993-1994)",
        "displayName-count-other": "Brasilianische Cruzeiro (1993-1994)",
        "symbol": "BRR"
    },
    "BRZ": {
        "displayName": "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-one": "Brasilianischer Cruzeiro (1942-1967)",
        "displayName-count-other": "Brasilianischer Cruzeiro (1942-
1967)",
        "symbol": "BRZ"
    }

```

```

    },
    "BSD": {
      "displayName": "Bahamas-Dollar",
      "displayName-count-one": "Bahamas-Dollar",
      "displayName-count-other": "Bahamas-Dollar",
      "symbol": "BSD",
      "symbol-alt-narrow": "$"
    },
    "BTN": {
      "displayName": "Bhutan-Ngultrum",
      "displayName-count-one": "Bhutan-Ngultrum",
      "displayName-count-other": "Bhutan-Ngultrum",
      "symbol": "BTN"
    },
    "BUK": {
      "displayName": "Birmanischer Kyat",
      "displayName-count-one": "Birmanischer Kyat",
      "displayName-count-other": "Birmanische Kyat",
      "symbol": "BUK"
    },
    "BWP": {
      "displayName": "Botswanischer Pula",
      "displayName-count-one": "Botswanischer Pula",
      "displayName-count-other": "Botswanische Pula",
      "symbol": "BWP",
      "symbol-alt-narrow": "P"
    },
    "BYB": {
      "displayName": "Belarus-Rubel (1994-1999)",
      "displayName-count-one": "Belarus-Rubel (1994-1999)",
      "displayName-count-other": "Belarus-Rubel (1994-1999)",
      "symbol": "BYB"
    },
    "BYN": {
      "displayName": "Weißrussischer Rubel",
      "displayName-count-one": "Weißrussischer Rubel",
      "displayName-count-other": "Weißrussische Rubel",
      "symbol": "BYN",
      "symbol-alt-narrow": "p."
    },
    "BYR": {
      "displayName": "Weißrussischer Rubel (2000-2016)",
      "displayName-count-one": "Weißrussischer Rubel (2000-2016)",
      "displayName-count-other": "Weißrussische Rubel (2000-2016)",
      "symbol": "BYR"
    },
    "BZD": {
      "displayName": "Belize-Dollar",
      "displayName-count-one": "Belize-Dollar",
      "displayName-count-other": "Belize-Dollar",
      "symbol": "BZD",
      "symbol-alt-narrow": "$"
    },
    "CAD": {
      "displayName": "Kanadischer Dollar",
      "displayName-count-one": "Kanadischer Dollar",
      "displayName-count-other": "Kanadische Dollar",

```



```
    "symbol": "CA$",
    "symbol-alt-narrow": "$"
  },
  "CDF": {
    "displayName": "Kongo-Franc",
    "displayName-count-one": "Kongo-Franc",
    "displayName-count-other": "Kongo-Francs",
    "symbol": "CDF"
  },
  "CHE": {
    "displayName": "WIR-Euro",
    "displayName-count-one": "WIR-Euro",
    "displayName-count-other": "WIR-Euro",
    "symbol": "CHE"
  },
  "CHF": {
    "displayName": "Schweizer Franken",
    "displayName-count-one": "Schweizer Franken",
    "displayName-count-other": "Schweizer Franken",
    "symbol": "CHF"
  },
  "CHW": {
    "displayName": "WIR Franken",
    "displayName-count-one": "WIR Franken",
    "displayName-count-other": "WIR Franken",
    "symbol": "CHW"
  },
  "CLE": {
    "displayName": "Chilenischer Escudo",
    "displayName-count-one": "Chilenischer Escudo",
    "displayName-count-other": "Chilenische Escudo",
    "symbol": "CLE"
  },
  "CLF": {
    "displayName": "Chilenische Unidades de Fomento",
    "displayName-count-one": "Chilenische Unidades de Fomento",
    "displayName-count-other": "Chilenische Unidades de Fomento",
    "symbol": "CLF"
  },
  "CLP": {
    "displayName": "Chilenischer Peso",
    "displayName-count-one": "Chilenischer Peso",
    "displayName-count-other": "Chilenische Pesos",
    "symbol": "CLP",
    "symbol-alt-narrow": "$"
  },
  "CNX": {
    "displayName": "Dollar der Chinesischen Volksbank",
    "displayName-count-one": "Dollar der Chinesischen Volksbank",
    "displayName-count-other": "Dollar der Chinesischen Volksbank",
    "symbol": "CNX"
  },
  "CNY": {
    "displayName": "Renminbi Yuan",
    "displayName-count-one": "Chinesischer Yuan",
    "displayName-count-other": "Renminbi Yuan",
    "symbol": "CN¥",
```

```

        "symbol-alt-narrow": "¥"
    },
    "COP": {
        "displayName": "Kolumbianischer Peso",
        "displayName-count-one": "Kolumbianischer Peso",
        "displayName-count-other": "Kolumbianische Pesos",
        "symbol": "COP",
        "symbol-alt-narrow": "$"
    },
    "COU": {
        "displayName": "Kolumbianische Unidades de valor real",
        "displayName-count-one": "Kolumbianische Unidad de valor real",
        "displayName-count-other": "Kolumbianische Unidades de valor
real",
        "symbol": "COU"
    },
    "CRC": {
        "displayName": "Costa-Rica-Colón",
        "displayName-count-one": "Costa-Rica-Colón",
        "displayName-count-other": "Costa-Rica-Colón",
        "symbol": "CRC",
        "symbol-alt-narrow": "₡"
    },
    "CSD": {
        "displayName": "Serbischer Dinar (2002-2006)",
        "displayName-count-one": "Serbischer Dinar (2002-2006)",
        "displayName-count-other": "Serbische Dinar (2002-2006)",
        "symbol": "CSD"
    },
    "CSK": {
        "displayName": "Tschechoslowakische Krone",
        "displayName-count-one": "Tschechoslowakische Kronen",
        "displayName-count-other": "Tschechoslowakische Kronen",
        "symbol": "CSK"
    },
    "CUC": {
        "displayName": "Kubanischer Peso (konvertibel)",
        "displayName-count-one": "Kubanischer Peso (konvertibel)",
        "displayName-count-other": "Kubanische Pesos (konvertibel)",
        "symbol": "CUC",
        "symbol-alt-narrow": "Cub$"
    },
    "CUP": {
        "displayName": "Kubanischer Peso",
        "displayName-count-one": "Kubanischer Peso",
        "displayName-count-other": "Kubanische Pesos",
        "symbol": "CUP",
        "symbol-alt-narrow": "$"
    },
    "CVE": {
        "displayName": "Cabo-Verde-Escudo",
        "displayName-count-one": "Cabo-Verde-Escudo",
        "displayName-count-other": "Cabo-Verde-Escudos",
        "symbol": "CVE"
    },
    "CYP": {
        "displayName": "Zypern-Pfund",

```

```

        "displayName-count-one": "Zypern Pfund",
        "displayName-count-other": "Zypern Pfund",
        "symbol": "CYP"
    },
    "CZK": {
        "displayName": "Tschechische Krone",
        "displayName-count-one": "Tschechische Krone",
        "displayName-count-other": "Tschechische Kronen",
        "symbol": "CZK",
        "symbol-alt-narrow": "Kč"
    },
    "DDM": {
        "displayName": "Mark der DDR",
        "displayName-count-one": "Mark der DDR",
        "displayName-count-other": "Mark der DDR",
        "symbol": "DDM"
    },
    "DEM": {
        "displayName": "Deutsche Mark",
        "displayName-count-one": "Deutsche Mark",
        "displayName-count-other": "Deutsche Mark",
        "symbol": "DM"
    },
    "DJF": {
        "displayName": "Dschibuti-Franc",
        "displayName-count-one": "Dschibuti-Franc",
        "displayName-count-other": "Dschibuti-Franc",
        "symbol": "DJF"
    },
    "DKK": {
        "displayName": "Dänische Krone",
        "displayName-count-one": "Dänische Krone",
        "displayName-count-other": "Dänische Kronen",
        "symbol": "DKK",
        "symbol-alt-narrow": "kr"
    },
    "DOP": {
        "displayName": "Dominikanischer Peso",
        "displayName-count-one": "Dominikanischer Peso",
        "displayName-count-other": "Dominikanische Pesos",
        "symbol": "DOP",
        "symbol-alt-narrow": "$"
    },
    "DZD": {
        "displayName": "Algerischer Dinar",
        "displayName-count-one": "Algerischer Dinar",
        "displayName-count-other": "Algerische Dinar",
        "symbol": "DZD"
    },
    "ECS": {
        "displayName": "Ecuadorianischer Sucre",
        "displayName-count-one": "Ecuadorianischer Sucre",
        "displayName-count-other": "Ecuadorianische Sucre",
        "symbol": "ECS"
    },
    "ECV": {
        "displayName": "Verrechnungseinheit für Ecuador",

```

```

    "displayName-count-one": "Verrechnungseinheiten für Ecuador",
    "displayName-count-other": "Verrechnungseinheiten für Ecuador",
    "symbol": "ECV"
  },
  "EEK": {
    "displayName": "Estnische Krone",
    "displayName-count-one": "Estnische Krone",
    "displayName-count-other": "Estnische Kronen",
    "symbol": "EEK"
  },
  "EGP": {
    "displayName": "Ägyptisches Pfund",
    "displayName-count-one": "Ägyptisches Pfund",
    "displayName-count-other": "Ägyptische Pfund",
    "symbol": "EGP",
    "symbol-alt-narrow": "E£"
  },
  "ERN": {
    "displayName": "Eritreischer Nakfa",
    "displayName-count-one": "Eritreischer Nakfa",
    "displayName-count-other": "Eritreische Nakfa",
    "symbol": "ERN"
  },
  "ESA": {
    "displayName": "Spanische Peseta (A-Konten)",
    "displayName-count-one": "Spanische Peseta (A-Konten)",
    "displayName-count-other": "Spanische Peseten (A-Konten)",
    "symbol": "ESA"
  },
  "ESB": {
    "displayName": "Spanische Peseta (konvertibel)",
    "displayName-count-one": "Spanische Peseta (konvertibel)",
    "displayName-count-other": "Spanische Peseten (konvertibel)",
    "symbol": "ESB"
  },
  "ESP": {
    "displayName": "Spanische Peseta",
    "displayName-count-one": "Spanische Peseta",
    "displayName-count-other": "Spanische Peseten",
    "symbol": "ESP",
    "symbol-alt-narrow": "₧"
  },
  "ETB": {
    "displayName": "Äthiopischer Birr",
    "displayName-count-one": "Äthiopischer Birr",
    "displayName-count-other": "Äthiopische Birr",
    "symbol": "ETB"
  },
  "EUR": {
    "displayName": "Euro",
    "displayName-count-one": "Euro",
    "displayName-count-other": "Euro",
    "symbol": "€",
    "symbol-alt-narrow": "€"
  },
  "FIM": {
    "displayName": "Finnische Mark",

```

```

        "displayName-count-one": "Finnische Mark",
        "displayName-count-other": "Finnische Mark",
        "symbol": "FIM"
    },
    "FJD": {
        "displayName": "Fidschi-Dollar",
        "displayName-count-one": "Fidschi-Dollar",
        "displayName-count-other": "Fidschi-Dollar",
        "symbol": "FJD",
        "symbol-alt-narrow": "$"
    },
    "FKP": {
        "displayName": "Falkland-Pfund",
        "displayName-count-one": "Falkland-Pfund",
        "displayName-count-other": "Falkland-Pfund",
        "symbol": "FKP",
        "symbol-alt-narrow": "Fl£"
    },
    "FRF": {
        "displayName": "Französischer Franc",
        "displayName-count-one": "Französischer Franc",
        "displayName-count-other": "Französische Franc",
        "symbol": "FRF"
    },
    "GBP": {
        "displayName": "Britisches Pfund",
        "displayName-count-one": "Britisches Pfund",
        "displayName-count-other": "Britische Pfund",
        "symbol": "£",
        "symbol-alt-narrow": "£"
    },
    "GEK": {
        "displayName": "Georgischer Kupon Larit",
        "displayName-count-one": "Georgischer Kupon Larit",
        "displayName-count-other": "Georgische Kupon Larit",
        "symbol": "GEK"
    },
    "GEL": {
        "displayName": "Georgischer Lari",
        "displayName-count-one": "Georgischer Lari",
        "displayName-count-other": "Georgische Lari",
        "symbol": "GEL",
        "symbol-alt-narrow": "ლ",
        "symbol-alt-variant": "ლ"
    },
    "GHC": {
        "displayName": "Ghanaischer Cedi (1979-2007)",
        "displayName-count-one": "Ghanaischer Cedi (1979-2007)",
        "displayName-count-other": "Ghanaische Cedi (1979-2007)",
        "symbol": "GHC"
    },
    "GHS": {
        "displayName": "Ghanaischer Cedi",
        "displayName-count-one": "Ghanaischer Cedi",
        "displayName-count-other": "Ghanaische Cedi",
        "symbol": "GHS"
    },
    },

```

```
"GIP": {
  "displayName": "Gibraltar-Pfund",
  "displayName-count-one": "Gibraltar-Pfund",
  "displayName-count-other": "Gibraltar Pfund",
  "symbol": "GIP",
  "symbol-alt-narrow": "£"
},
"GMD": {
  "displayName": "Gambia-Dalasi",
  "displayName-count-one": "Gambia-Dalasi",
  "displayName-count-other": "Gambia-Dalasi",
  "symbol": "GMD"
},
"GNF": {
  "displayName": "Guinea-Franc",
  "displayName-count-one": "Guinea-Franc",
  "displayName-count-other": "Guinea-Franc",
  "symbol": "GNF",
  "symbol-alt-narrow": "F.G."
},
"GNS": {
  "displayName": "Guineischer Syli",
  "displayName-count-one": "Guineischer Syli",
  "displayName-count-other": "Guineische Syli",
  "symbol": "GNS"
},
"GQE": {
  "displayName": "Äquatorialguinea-Ekwele",
  "displayName-count-one": "Äquatorialguinea-Ekwele",
  "displayName-count-other": "Äquatorialguinea-Ekwele",
  "symbol": "GQE"
},
"GRD": {
  "displayName": "Griechische Drachme",
  "displayName-count-one": "Griechische Drachme",
  "displayName-count-other": "Griechische Drachmen",
  "symbol": "GRD"
},
"GTQ": {
  "displayName": "Guatemaltekinscher Quetzal",
  "displayName-count-one": "Guatemaltekinscher Quetzal",
  "displayName-count-other": "Guatemaltekinsche Quetzales",
  "symbol": "GTQ",
  "symbol-alt-narrow": "Q"
},
"GWE": {
  "displayName": "Portugiesisch Guinea Escudo",
  "displayName-count-one": "Portugiesisch Guinea Escudo",
  "displayName-count-other": "Portugiesisch Guinea Escudo",
  "symbol": "GWE"
},
"GWP": {
  "displayName": "Guinea-Bissau Peso",
  "displayName-count-one": "Guinea-Bissau Peso",
  "displayName-count-other": "Guinea-Bissau Pesos",
  "symbol": "GWP"
},
```

```

"GYD": {
  "displayName": "Guyana-Dollar",
  "displayName-count-one": "Guyana-Dollar",
  "displayName-count-other": "Guyana-Dollar",
  "symbol": "GYD",
  "symbol-alt-narrow": "$"
},
"HKD": {
  "displayName": "Hongkong-Dollar",
  "displayName-count-one": "Hongkong-Dollar",
  "displayName-count-other": "Hongkong-Dollar",
  "symbol": "HK$",
  "symbol-alt-narrow": "$"
},
"HNL": {
  "displayName": "Honduras-Lempira",
  "displayName-count-one": "Honduras-Lempira",
  "displayName-count-other": "Honduras-Lempira",
  "symbol": "HNL",
  "symbol-alt-narrow": "L"
},
"HRD": {
  "displayName": "Kroatischer Dinar",
  "displayName-count-one": "Kroatischer Dinar",
  "displayName-count-other": "Kroatische Dinar",
  "symbol": "HRD"
},
"HRK": {
  "displayName": "Kroatischer Kuna",
  "displayName-count-one": "Kroatischer Kuna",
  "displayName-count-other": "Kroatische Kuna",
  "symbol": "HRK",
  "symbol-alt-narrow": "kn"
},
"HTG": {
  "displayName": "Haitianische Gourde",
  "displayName-count-one": "Haitianische Gourde",
  "displayName-count-other": "Haitianische Gourdes",
  "symbol": "HTG"
},
"HUF": {
  "displayName": "Ungarischer Forint",
  "displayName-count-one": "Ungarischer Forint",
  "displayName-count-other": "Ungarische Forint",
  "symbol": "HUF",
  "symbol-alt-narrow": "Ft"
},
"IDR": {
  "displayName": "Indonesische Rupiah",
  "displayName-count-one": "Indonesische Rupiah",
  "displayName-count-other": "Indonesische Rupiah",
  "symbol": "IDR",
  "symbol-alt-narrow": "Rp"
},
"IEP": {
  "displayName": "Irisches Pfund",
  "displayName-count-one": "Irisches Pfund",

```

```
    "displayName-count-other": "Irische Pfund",
    "symbol": "IEP"
  },
  "ILP": {
    "displayName": "Israelisches Pfund",
    "displayName-count-one": "Israelisches Pfund",
    "displayName-count-other": "Israelische Pfund",
    "symbol": "ILP"
  },
  "ILR": {
    "displayName": "Israelischer Schekel (1980-1985)",
    "displayName-count-one": "Israelischer Schekel (1980-1985)",
    "displayName-count-other": "Israelische Schekel (1980-1985)"
  },
  "ILS": {
    "displayName": "Israelischer Neuer Schekel",
    "displayName-count-one": "Israelischer Neuer Schekel",
    "displayName-count-other": "Israelische Neue Schekel",
    "symbol": "₪",
    "symbol-alt-narrow": "₪"
  },
  "INR": {
    "displayName": "Indische Rupie",
    "displayName-count-one": "Indische Rupie",
    "displayName-count-other": "Indische Rupien",
    "symbol": "₹",
    "symbol-alt-narrow": "₹"
  },
  "IQD": {
    "displayName": "Irakischer Dinar",
    "displayName-count-one": "Irakischer Dinar",
    "displayName-count-other": "Irakische Dinar",
    "symbol": "IQD"
  },
  "IRR": {
    "displayName": "Iranischer Rial",
    "displayName-count-one": "Iranischer Rial",
    "displayName-count-other": "Iranische Rial",
    "symbol": "IRR"
  },
  "ISJ": {
    "displayName": "Isländische Krone (1918-1981)",
    "displayName-count-one": "Isländische Krone (1918-1981)",
    "displayName-count-other": "Isländische Kronen (1918-1981)"
  },
  "ISK": {
    "displayName": "Isländische Krone",
    "displayName-count-one": "Isländische Krone",
    "displayName-count-other": "Isländische Kronen",
    "symbol": "ISK",
    "symbol-alt-narrow": "kr"
  },
  "ITL": {
    "displayName": "Italienische Lira",
    "displayName-count-one": "Italienische Lira",
    "displayName-count-other": "Italienische Lire",
    "symbol": "ITL"
```



```

    },
    "JMD": {
      "displayName": "Jamaika-Dollar",
      "displayName-count-one": "Jamaika-Dollar",
      "displayName-count-other": "Jamaika-Dollar",
      "symbol": "JMD",
      "symbol-alt-narrow": "$"
    },
    "JOD": {
      "displayName": "Jordanischer Dinar",
      "displayName-count-one": "Jordanischer Dinar",
      "displayName-count-other": "Jordanische Dinar",
      "symbol": "JOD"
    },
    "JPY": {
      "displayName": "Japanischer Yen",
      "displayName-count-one": "Japanischer Yen",
      "displayName-count-other": "Japanische Yen",
      "symbol": "¥",
      "symbol-alt-narrow": "¥"
    },
    "KES": {
      "displayName": "Kenia-Schilling",
      "displayName-count-one": "Kenia-Schilling",
      "displayName-count-other": "Kenia-Schilling",
      "symbol": "KES"
    },
    "KGS": {
      "displayName": "Kirgisischer Som",
      "displayName-count-one": "Kirgisischer Som",
      "displayName-count-other": "Kirgisische Som",
      "symbol": "KGS"
    },
    "KHR": {
      "displayName": "Kambodschanischer Riel",
      "displayName-count-one": "Kambodschanischer Riel",
      "displayName-count-other": "Kambodschanische Riel",
      "symbol": "KHR",
      "symbol-alt-narrow": "៛"
    },
    "KMF": {
      "displayName": "Komoren-Franc",
      "displayName-count-one": "Komoren-Franc",
      "displayName-count-other": "Komoren-Francs",
      "symbol": "KMF",
      "symbol-alt-narrow": "FC"
    },
    "KPW": {
      "displayName": "Nordkoreanischer Won",
      "displayName-count-one": "Nordkoreanischer Won",
      "displayName-count-other": "Nordkoreanische Won",
      "symbol": "KPW",
      "symbol-alt-narrow": "₩"
    },
    "KRH": {
      "displayName": "Südkoreanischer Hwan (1953-1962)",

```

```

    "displayName-count-one": "Südkoreanischer Hwan (1953-1962)",
    "displayName-count-other": "Südkoreanischer Hwan (1953-1962)",
    "symbol": "KRH"
  },
  "KRO": {
    "displayName": "Südkoreanischer Won (1945-1953)",
    "displayName-count-one": "Südkoreanischer Won (1945-1953)",
    "displayName-count-other": "Südkoreanischer Won (1945-1953)",
    "symbol": "KRO"
  },
  "KRW": {
    "displayName": "Südkoreanischer Won",
    "displayName-count-one": "Südkoreanischer Won",
    "displayName-count-other": "Südkoreanische Won",
    "symbol": "₩",
    "symbol-alt-narrow": "₩"
  },
  "KWD": {
    "displayName": "Kuwait-Dinar",
    "displayName-count-one": "Kuwait-Dinar",
    "displayName-count-other": "Kuwait-Dinar",
    "symbol": "KWD"
  },
  "KYD": {
    "displayName": "Kaiman-Dollar",
    "displayName-count-one": "Kaiman-Dollar",
    "displayName-count-other": "Kaiman-Dollar",
    "symbol": "KYD",
    "symbol-alt-narrow": "$"
  },
  "KZT": {
    "displayName": "Kasachischer Tenge",
    "displayName-count-one": "Kasachischer Tenge",
    "displayName-count-other": "Kasachische Tenge",
    "symbol": "KZT",
    "symbol-alt-narrow": "₸"
  },
  "LAK": {
    "displayName": "Laotischer Kip",
    "displayName-count-one": "Laotischer Kip",
    "displayName-count-other": "Laotische Kip",
    "symbol": "LAK",
    "symbol-alt-narrow": "₭"
  },
  "LBP": {
    "displayName": "Libanesisches Pfund",
    "displayName-count-one": "Libanesisches Pfund",
    "displayName-count-other": "Libanesische Pfund",
    "symbol": "LBP",
    "symbol-alt-narrow": "L£"
  },
  "LKR": {
    "displayName": "Sri-Lanka-Rupie",
    "displayName-count-one": "Sri-Lanka-Rupie",
    "displayName-count-other": "Sri-Lanka-Rupien",
    "symbol": "LKR",
    "symbol-alt-narrow": "Rs"
  }

```

```
    },
    "LRD": {
      "displayName": "Liberianischer Dollar",
      "displayName-count-one": "Liberianischer Dollar",
      "displayName-count-other": "Liberianische Dollar",
      "symbol": "LRD",
      "symbol-alt-narrow": "$"
    },
    "LSL": {
      "displayName": "Loti",
      "displayName-count-one": "Loti",
      "displayName-count-other": "Loti",
      "symbol": "LSL"
    },
    "LTL": {
      "displayName": "Litauischer Litas",
      "displayName-count-one": "Litauischer Litas",
      "displayName-count-other": "Litauische Litas",
      "symbol": "LTL",
      "symbol-alt-narrow": "Lt"
    },
    "LTT": {
      "displayName": "Litauischer Talonas",
      "displayName-count-one": "Litauische Talonas",
      "displayName-count-other": "Litauische Talonas",
      "symbol": "LTT"
    },
    "LUC": {
      "displayName": "Luxemburgischer Franc (konvertibel)",
      "displayName-count-one": "Luxemburgische Franc (konvertibel)",
      "displayName-count-other": "Luxemburgische Franc (konvertibel)",
      "symbol": "LUC"
    },
    "LUF": {
      "displayName": "Luxemburgischer Franc",
      "displayName-count-one": "Luxemburgische Franc",
      "displayName-count-other": "Luxemburgische Franc",
      "symbol": "LUF"
    },
    "LUL": {
      "displayName": "Luxemburgischer Finanz-Franc",
      "displayName-count-one": "Luxemburgische Finanz-Franc",
      "displayName-count-other": "Luxemburgische Finanz-Franc",
      "symbol": "LUL"
    },
    "LVL": {
      "displayName": "Lettischer Lats",
      "displayName-count-one": "Lettischer Lats",
      "displayName-count-other": "Lettische Lats",
      "symbol": "LVL",
      "symbol-alt-narrow": "Ls"
    },
    "LVR": {
      "displayName": "Lettischer Rubel",
      "displayName-count-one": "Lettische Rubel",
      "displayName-count-other": "Lettische Rubel",
      "symbol": "LVR"
    }
  }
```

```

    },
    "LYD": {
      "displayName": "Libyscher Dinar",
      "displayName-count-one": "Libyscher Dinar",
      "displayName-count-other": "Libysche Dinar",
      "symbol": "LYD"
    },
    "MAD": {
      "displayName": "Marokkanischer Dirham",
      "displayName-count-one": "Marokkanischer Dirham",
      "displayName-count-other": "Marokkanische Dirham",
      "symbol": "MAD"
    },
    "MAF": {
      "displayName": "Marokkanischer Franc",
      "displayName-count-one": "Marokkanische Franc",
      "displayName-count-other": "Marokkanische Franc",
      "symbol": "MAF"
    },
    "MCF": {
      "displayName": "Monegassischer Franc",
      "displayName-count-one": "Monegassischer Franc",
      "displayName-count-other": "Monegassische Franc",
      "symbol": "MCF"
    },
    "MDC": {
      "displayName": "Moldau-Cupon",
      "displayName-count-one": "Moldau-Cupon",
      "displayName-count-other": "Moldau-Cupon",
      "symbol": "MDC"
    },
    "MDL": {
      "displayName": "Moldau-Leu",
      "displayName-count-one": "Moldau-Leu",
      "displayName-count-other": "Moldau-Leu",
      "symbol": "MDL"
    },
    "MGA": {
      "displayName": "Madagaskar-Ariary",
      "displayName-count-one": "Madagaskar-Ariary",
      "displayName-count-other": "Madagaskar-Ariary",
      "symbol": "MGA",
      "symbol-alt-narrow": "Ar"
    },
    "MGF": {
      "displayName": "Madagaskar-Franc",
      "displayName-count-one": "Madagaskar-Franc",
      "displayName-count-other": "Madagaskar-Franc",
      "symbol": "MGF"
    },
    "MKD": {
      "displayName": "Mazedonischer Denar",
      "displayName-count-one": "Mazedonischer Denar",
      "displayName-count-other": "Mazedonische Denari",
      "symbol": "MKD"
    },
    "MKN": {

```

```

    "displayName": "Mazedonischer Denar (1992-1993)",
    "displayName-count-one": "Mazedonischer Denar (1992-1993)",
    "displayName-count-other": "Mazedonische Denar (1992-1993)",
    "symbol": "MKN"
  },
  "MLF": {
    "displayName": "Malischer Franc",
    "displayName-count-one": "Malische Franc",
    "displayName-count-other": "Malische Franc",
    "symbol": "MLF"
  },
  "MMK": {
    "displayName": "Myanmarischer Kyat",
    "displayName-count-one": "Myanmarischer Kyat",
    "displayName-count-other": "Myanmarische Kyat",
    "symbol": "MMK",
    "symbol-alt-narrow": "K"
  },
  "MNT": {
    "displayName": "Mongolischer Tögrög",
    "displayName-count-one": "Mongolischer Tögrög",
    "displayName-count-other": "Mongolische Tögrög",
    "symbol": "MNT",
    "symbol-alt-narrow": "₮"
  },
  "MOP": {
    "displayName": "Macao-Pataca",
    "displayName-count-one": "Macao-Pataca",
    "displayName-count-other": "Macao-Pataca",
    "symbol": "MOP"
  },
  "MRO": {
    "displayName": "Mauretanischer Ouguiya",
    "displayName-count-one": "Mauretanischer Ouguiya",
    "displayName-count-other": "Mauretanische Ouguiya",
    "symbol": "MRO"
  },
  "MTL": {
    "displayName": "Maltesische Lira",
    "displayName-count-one": "Maltesische Lira",
    "displayName-count-other": "Maltesische Lira",
    "symbol": "MTL"
  },
  "MTP": {
    "displayName": "Maltesisches Pfund",
    "displayName-count-one": "Maltesische Pfund",
    "displayName-count-other": "Maltesische Pfund",
    "symbol": "MTP"
  },
  "MUR": {
    "displayName": "Mauritius-Rupie",
    "displayName-count-one": "Mauritius-Rupie",
    "displayName-count-other": "Mauritius-Rupien",
    "symbol": "MUR",
    "symbol-alt-narrow": "Rs"
  },
  "MVP": {

```

```

        "displayName": "Malediven-Rupie (alt)",
        "displayName-count-one": "Malediven-Rupie (alt)",
        "displayName-count-other": "Malediven-Rupien (alt)"
    },
    "MVR": {
        "displayName": "Malediven-Rufiyaa",
        "displayName-count-one": "Malediven-Rufiyaa",
        "displayName-count-other": "Malediven-Rupien",
        "symbol": "MVR"
    },
    "MWK": {
        "displayName": "Malawi-Kwacha",
        "displayName-count-one": "Malawi-Kwacha",
        "displayName-count-other": "Malawi-Kwacha",
        "symbol": "MWK"
    },
    "MXN": {
        "displayName": "Mexikanischer Peso",
        "displayName-count-one": "Mexikanischer Peso",
        "displayName-count-other": "Mexikanische Pesos",
        "symbol": "MX$",
        "symbol-alt-narrow": "$"
    },
    "MXP": {
        "displayName": "Mexikanischer Silber-Peso (1861-1992)",
        "displayName-count-one": "Mexikanische Silber-Peso (1861-1992)",
        "displayName-count-other": "Mexikanische Silber-Pesos (1861-
1992)",
        "symbol": "MXP"
    },
    "MXV": {
        "displayName": "Mexicanischer Unidad de Inversion (UDI)",
        "displayName-count-one": "Mexicanischer Unidad de Inversion
(UDI)",
        "displayName-count-other": "Mexikanische Unidad de Inversion
(UDI)",
        "symbol": "MXV"
    },
    "MYR": {
        "displayName": "Malaysischer Ringgit",
        "displayName-count-one": "Malaysischer Ringgit",
        "displayName-count-other": "Malaysische Ringgit",
        "symbol": "MYR",
        "symbol-alt-narrow": "RM"
    },
    "MZE": {
        "displayName": "Mosambikanischer Escudo",
        "displayName-count-one": "Mozambikanische Escudo",
        "displayName-count-other": "Mozambikanische Escudo",
        "symbol": "MZE"
    },
    "MZM": {
        "displayName": "Mosambikanischer Metical (1980-2006)",
        "displayName-count-one": "Mosambikanischer Metical (1980-2006)",
        "displayName-count-other": "Mosambikanische Meticais (1980-
2006)",
        "symbol": "MZM"
    }

```

```

    },
    "MZN": {
      "displayName": "Mosambikanischer Metical",
      "displayName-count-one": "Mosambikanischer Metical",
      "displayName-count-other": "Mosambikanische Meticais",
      "symbol": "MZN"
    },
    "NAD": {
      "displayName": "Namibia-Dollar",
      "displayName-count-one": "Namibia-Dollar",
      "displayName-count-other": "Namibia-Dollar",
      "symbol": "NAD",
      "symbol-alt-narrow": "$"
    },
    "NGN": {
      "displayName": "Nigerianischer Naira",
      "displayName-count-one": "Nigerianischer Naira",
      "displayName-count-other": "Nigerianische Naira",
      "symbol": "NGN",
      "symbol-alt-narrow": "₦"
    },
    "NIC": {
      "displayName": "Nicaraguanischer Córdoba (1988-1991)",
      "displayName-count-one": "Nicaraguanischer Córdoba (1988-1991)",
      "displayName-count-other": "Nicaraguanische Córdoba (1988-1991)",
      "symbol": "NIC"
    },
    "NIO": {
      "displayName": "Nicaragua-Córdoba",
      "displayName-count-one": "Nicaragua-Córdoba",
      "displayName-count-other": "Nicaragua-Córdobas",
      "symbol": "NIO",
      "symbol-alt-narrow": "C$"
    },
    "NLG": {
      "displayName": "Niederländischer Gulden",
      "displayName-count-one": "Niederländischer Gulden",
      "displayName-count-other": "Niederländische Gulden",
      "symbol": "NLG"
    },
    "NOK": {
      "displayName": "Norwegische Krone",
      "displayName-count-one": "Norwegische Krone",
      "displayName-count-other": "Norwegische Kronen",
      "symbol": "NOK",
      "symbol-alt-narrow": "kr"
    },
    "NPR": {
      "displayName": "Nepalesische Rupie",
      "displayName-count-one": "Nepalesische Rupie",
      "displayName-count-other": "Nepalesische Rupien",
      "symbol": "NPR",
      "symbol-alt-narrow": "Rs"
    },
    "NZD": {
      "displayName": "Neuseeland-Dollar",
      "displayName-count-one": "Neuseeland-Dollar",

```

```

        "displayName-count-other": "Neuseeland-Dollar",
        "symbol": "NZ$",
        "symbol-alt-narrow": "$"
    },
    "OMR": {
        "displayName": "Omanischer Rial",
        "displayName-count-one": "Omanischer Rial",
        "displayName-count-other": "Omanische Rials",
        "symbol": "OMR"
    },
    "PAB": {
        "displayName": "Panamaischer Balboa",
        "displayName-count-one": "Panamaischer Balboa",
        "displayName-count-other": "Panamaische Balboas",
        "symbol": "PAB"
    },
    "PEI": {
        "displayName": "Peruanischer Inti",
        "displayName-count-one": "Peruanische Inti",
        "displayName-count-other": "Peruanische Inti",
        "symbol": "PEI"
    },
    "PEN": {
        "displayName": "Peruanischer Sol",
        "displayName-count-one": "Peruanischer Sol",
        "displayName-count-other": "Peruanische Sol",
        "symbol": "PEN"
    },
    "PES": {
        "displayName": "Peruanischer Sol (1863-1965)",
        "displayName-count-one": "Peruanischer Sol (1863-1965)",
        "displayName-count-other": "Peruanische Sol (1863-1965)",
        "symbol": "PES"
    },
    "PGK": {
        "displayName": "Papua-Neuguineischer Kina",
        "displayName-count-one": "Papua-Neuguineischer Kina",
        "displayName-count-other": "Papua-Neuguineische Kina",
        "symbol": "PGK"
    },
    "PHP": {
        "displayName": "Philippinischer Peso",
        "displayName-count-one": "Philippinischer Peso",
        "displayName-count-other": "Philippinische Pesos",
        "symbol": "PHP",
        "symbol-alt-narrow": "₱"
    },
    "PKR": {
        "displayName": "Pakistanische Rupie",
        "displayName-count-one": "Pakistanische Rupie",
        "displayName-count-other": "Pakistanische Rupien",
        "symbol": "PKR",
        "symbol-alt-narrow": "Rs"
    },
    "PLN": {
        "displayName": "Polnischer Złoty",
        "displayName-count-one": "Polnischer Złoty",

```



```

    "displayName-count-other": "Polnische Złoty",
    "symbol": "PLN",
    "symbol-alt-narrow": "zł"
  },
  "PLZ": {
    "displayName": "Polnischer Zloty (1950-1995)",
    "displayName-count-one": "Polnischer Zloty (1950-1995)",
    "displayName-count-other": "Polnische Zloty (1950-1995)",
    "symbol": "PLZ"
  },
  "PTE": {
    "displayName": "Portugiesischer Escudo",
    "displayName-count-one": "Portugiesische Escudo",
    "displayName-count-other": "Portugiesische Escudo",
    "symbol": "PTE"
  },
  "PYG": {
    "displayName": "Paraguayischer Guaraní",
    "displayName-count-one": "Paraguayischer Guaraní",
    "displayName-count-other": "Paraguayische Guaraníes",
    "symbol": "PYG",
    "symbol-alt-narrow": "₲"
  },
  "QAR": {
    "displayName": "Katar-Riyal",
    "displayName-count-one": "Katar-Riyal",
    "displayName-count-other": "Katar-Riyal",
    "symbol": "QAR"
  },
  "RHD": {
    "displayName": "Rhodesischer Dollar",
    "displayName-count-one": "Rhodesische Dollar",
    "displayName-count-other": "Rhodesische Dollar",
    "symbol": "RHD"
  },
  "ROL": {
    "displayName": "Rumänischer Leu (1952-2006)",
    "displayName-count-one": "Rumänischer Leu (1952-2006)",
    "displayName-count-other": "Rumänische Leu (1952-2006)",
    "symbol": "ROL"
  },
  "RON": {
    "displayName": "Rumänischer Leu",
    "displayName-count-one": "Rumänischer Leu",
    "displayName-count-other": "Rumänische Leu",
    "symbol": "RON",
    "symbol-alt-narrow": "L"
  },
  "RSD": {
    "displayName": "Serbischer Dinar",
    "displayName-count-one": "Serbischer Dinar",
    "displayName-count-other": "Serbische Dinaren",
    "symbol": "RSD"
  },
  "RUB": {
    "displayName": "Russischer Rubel",
    "displayName-count-one": "Russischer Rubel",

```

```

        "displayName-count-other": "Russische Rubel",
        "symbol": "RUB",
        "symbol-alt-narrow": "₽"
    },
    "RUR": {
        "displayName": "Russischer Rubel (1991-1998)",
        "displayName-count-one": "Russischer Rubel (1991-1998)",
        "displayName-count-other": "Russische Rubel (1991-1998)",
        "symbol": "RUR",
        "symbol-alt-narrow": "p."
    },
    "RWF": {
        "displayName": "Ruanda-Franc",
        "displayName-count-one": "Ruanda-Franc",
        "displayName-count-other": "Ruanda-Francs",
        "symbol": "RWF",
        "symbol-alt-narrow": "F.Rw"
    },
    "SAR": {
        "displayName": "Saudi-Rial",
        "displayName-count-one": "Saudi-Rial",
        "displayName-count-other": "Saudi-Rial",
        "symbol": "SAR"
    },
    "SBD": {
        "displayName": "Salomonen-Dollar",
        "displayName-count-one": "Salomonen-Dollar",
        "displayName-count-other": "Salomonen-Dollar",
        "symbol": "SBD",
        "symbol-alt-narrow": "$"
    },
    "SCR": {
        "displayName": "Seychellen-Rupie",
        "displayName-count-one": "Seychellen-Rupie",
        "displayName-count-other": "Seychellen-Rupien",
        "symbol": "SCR"
    },
    "SDD": {
        "displayName": "Sudanesischer Dinar (1992-2007)",
        "displayName-count-one": "Sudanesischer Dinar (1992-2007)",
        "displayName-count-other": "Sudanesische Dinar (1992-2007)",
        "symbol": "SDD"
    },
    "SDG": {
        "displayName": "Sudanesisches Pfund",
        "displayName-count-one": "Sudanesisches Pfund",
        "displayName-count-other": "Sudanesische Pfund",
        "symbol": "SDG"
    },
    "SDP": {
        "displayName": "Sudanesisches Pfund (1957-1998)",
        "displayName-count-one": "Sudanesisches Pfund (1957-1998)",
        "displayName-count-other": "Sudanesische Pfund (1957-1998)",
        "symbol": "SDP"
    },
    "SEK": {
        "displayName": "Schwedische Krone",

```

```

        "displayName-count-one": "Schwedische Krone",
        "displayName-count-other": "Schwedische Kronen",
        "symbol": "SEK",
        "symbol-alt-narrow": "kr"
    },
    "SGD": {
        "displayName": "Singapur-Dollar",
        "displayName-count-one": "Singapur-Dollar",
        "displayName-count-other": "Singapur-Dollar",
        "symbol": "SGD",
        "symbol-alt-narrow": "$"
    },
    "SHP": {
        "displayName": "St. Helena-Pfund",
        "displayName-count-one": "St. Helena-Pfund",
        "displayName-count-other": "St. Helena-Pfund",
        "symbol": "SHP",
        "symbol-alt-narrow": "£"
    },
    "SIT": {
        "displayName": "Slowenischer Tolar",
        "displayName-count-one": "Slowenischer Tolar",
        "displayName-count-other": "Slowenische Tolar",
        "symbol": "SIT"
    },
    "SKK": {
        "displayName": "Slowakische Krone",
        "displayName-count-one": "Slowakische Kronen",
        "displayName-count-other": "Slowakische Kronen",
        "symbol": "SKK"
    },
    "SLL": {
        "displayName": "Sierra-leonischer Leone",
        "displayName-count-one": "Sierra-leonischer Leone",
        "displayName-count-other": "Sierra-leonische Leones",
        "symbol": "SLL"
    },
    "SOS": {
        "displayName": "Somalia-Schilling",
        "displayName-count-one": "Somalia-Schilling",
        "displayName-count-other": "Somalia-Schilling",
        "symbol": "SOS"
    },
    "SRD": {
        "displayName": "Suriname-Dollar",
        "displayName-count-one": "Suriname-Dollar",
        "displayName-count-other": "Suriname-Dollar",
        "symbol": "SRD",
        "symbol-alt-narrow": "$"
    },
    "SRG": {
        "displayName": "Suriname Gulden",
        "displayName-count-one": "Suriname-Gulden",
        "displayName-count-other": "Suriname-Gulden",
        "symbol": "SRG"
    },
    "SSP": {

```

```

        "displayName": "Südsudanesisches Pfund",
        "displayName-count-one": "Südsudanesisches Pfund",
        "displayName-count-other": "Südsudanesische Pfund",
        "symbol": "SSP",
        "symbol-alt-narrow": "£"
    },
    "STD": {
        "displayName": "São-toméischer Dobra",
        "displayName-count-one": "São-toméischer Dobra",
        "displayName-count-other": "São-toméische Dobra",
        "symbol": "STD",
        "symbol-alt-narrow": "Db"
    },
    "SUR": {
        "displayName": "Sowjetischer Rubel",
        "displayName-count-one": "Sowjetische Rubel",
        "displayName-count-other": "Sowjetische Rubel",
        "symbol": "SUR"
    },
    "SVC": {
        "displayName": "El Salvador Colon",
        "displayName-count-one": "El Salvador-Colon",
        "displayName-count-other": "El Salvador-Colon",
        "symbol": "SVC"
    },
    "SYP": {
        "displayName": "Syrisches Pfund",
        "displayName-count-one": "Syrisches Pfund",
        "displayName-count-other": "Syrische Pfund",
        "symbol": "SYP",
        "symbol-alt-narrow": "SYP"
    },
    "SZL": {
        "displayName": "Swasiländischer Lilangeni",
        "displayName-count-one": "Swasiländischer Lilangeni",
        "displayName-count-other": "Swasiländische Emalangeni",
        "symbol": "SZL"
    },
    "THB": {
        "displayName": "Thailändischer Baht",
        "displayName-count-one": "Thailändischer Baht",
        "displayName-count-other": "Thailändische Baht",
        "symbol": "฿",
        "symbol-alt-narrow": "฿"
    },
    "TJR": {
        "displayName": "Tadschikistan Rubel",
        "displayName-count-one": "Tadschikistan-Rubel",
        "displayName-count-other": "Tadschikistan-Rubel",
        "symbol": "TJR"
    },
    "TJS": {
        "displayName": "Tadschikistan-Somoni",
        "displayName-count-one": "Tadschikistan-Somoni",
        "displayName-count-other": "Tadschikistan-Somoni",
        "symbol": "TJS"
    },
    },

```

```
"TMM": {
  "displayName": "Turkmenistan-Manat (1993-2009)",
  "displayName-count-one": "Turkmenistan-Manat (1993-2009)",
  "displayName-count-other": "Turkmenistan-Manat (1993-2009)",
  "symbol": "TMM"
},
"TMT": {
  "displayName": "Turkmenistan-Manat",
  "displayName-count-one": "Turkmenistan-Manat",
  "displayName-count-other": "Turkmenistan-Manat",
  "symbol": "TMT"
},
"TND": {
  "displayName": "Tunesischer Dinar",
  "displayName-count-one": "Tunesischer Dinar",
  "displayName-count-other": "Tunesische Dinar",
  "symbol": "TND"
},
"TOP": {
  "displayName": "Tongaischer Pa'anga",
  "displayName-count-one": "Tongaischer Pa'anga",
  "displayName-count-other": "Tongaische Pa'anga",
  "symbol": "TOP",
  "symbol-alt-narrow": "T$"
},
"TPE": {
  "displayName": "Timor-Escudo",
  "displayName-count-one": "Timor-Escudo",
  "displayName-count-other": "Timor-Escudo",
  "symbol": "TPE"
},
"TRL": {
  "displayName": "Türkische Lira (1922-2005)",
  "displayName-count-one": "Türkische Lira (1922-2005)",
  "displayName-count-other": "Türkische Lira (1922-2005)",
  "symbol": "TRL"
},
"TRY": {
  "displayName": "Türkische Lira",
  "displayName-count-one": "Türkische Lira",
  "displayName-count-other": "Türkische Lira",
  "symbol": "TRY",
  "symbol-alt-narrow": "₺",
  "symbol-alt-variant": "TL"
},
"TTD": {
  "displayName": "Trinidad und Tobago-Dollar",
  "displayName-count-one": "Trinidad und Tobago-Dollar",
  "displayName-count-other": "Trinidad und Tobago-Dollar",
  "symbol": "TTD",
  "symbol-alt-narrow": "$"
},
"PWD": {
  "displayName": "Palau-Dollar",
  "displayName-count-one": "Palau-Dollar",
  "displayName-count-other": "Palau-Dollar",
  "symbol": "PWD",
  "symbol-alt-narrow": "$"
},
"TWB": {
  "displayName": "Neuer Taiwan-Dollar",
  "displayName-count-one": "Neuer Taiwan-Dollar",
  "displayName-count-other": "Neue Taiwan-Dollar",
  "symbol": "NT$"
```

```

        "symbol-alt-narrow": "NT$"
    },
    "TZS": {
        "displayName": "Tansania-Schilling",
        "displayName-count-one": "Tansania-Schilling",
        "displayName-count-other": "Tansania-Schilling",
        "symbol": "TZS"
    },
    "UAH": {
        "displayName": "Ukrainische Hrywnja",
        "displayName-count-one": "Ukrainische Hrywnja",
        "displayName-count-other": "Ukrainische Hrywen",
        "symbol": "UAH",
        "symbol-alt-narrow": "₴"
    },
    "UAK": {
        "displayName": "Ukrainischer Karbovanetz",
        "displayName-count-one": "Ukrainische Karbovanetz",
        "displayName-count-other": "Ukrainische Karbovanetz",
        "symbol": "UAK"
    },
    "UGS": {
        "displayName": "Uganda-Schilling (1966-1987)",
        "displayName-count-one": "Uganda-Schilling (1966-1987)",
        "displayName-count-other": "Uganda-Schilling (1966-1987)",
        "symbol": "UGS"
    },
    "UGX": {
        "displayName": "Uganda-Schilling",
        "displayName-count-one": "Uganda-Schilling",
        "displayName-count-other": "Uganda-Schilling",
        "symbol": "UGX"
    },
    "USD": {
        "displayName": "US-Dollar",
        "displayName-count-one": "US-Dollar",
        "displayName-count-other": "US-Dollar",
        "symbol": "$",
        "symbol-alt-narrow": "$"
    },
    "USN": {
        "displayName": "US Dollar (Nächster Tag)",
        "displayName-count-one": "US-Dollar (Nächster Tag)",
        "displayName-count-other": "US-Dollar (Nächster Tag)",
        "symbol": "USN"
    },
    "USS": {
        "displayName": "US Dollar (Gleicher Tag)",
        "displayName-count-one": "US-Dollar (Gleicher Tag)",
        "displayName-count-other": "US-Dollar (Gleicher Tag)",
        "symbol": "USS"
    },
    "UYI": {
        "displayName": "Uruguayischer Peso (Indexierte
Rechnungseinheiten)",
        "displayName-count-one": "Uruguayischer Peso (Indexierte
Rechnungseinheiten)",

```

```

        "displayName-count-other": "Uruguayische Pesos (Indexierte
Rechnungseinheiten)",
        "symbol": "UYI"
    },
    "UYP": {
        "displayName": "Uruguayischer Peso (1975-1993)",
        "displayName-count-one": "Uruguayischer Peso (1975-1993)",
        "displayName-count-other": "Uruguayische Pesos (1975-1993)",
        "symbol": "UYP"
    },
    "UYU": {
        "displayName": "Uruguayischer Peso",
        "displayName-count-one": "Uruguayischer Peso",
        "displayName-count-other": "Uruguayische Pesos",
        "symbol": "UYU",
        "symbol-alt-narrow": "$"
    },
    "UZS": {
        "displayName": "Usbekistan-Sum",
        "displayName-count-one": "Usbekistan-Sum",
        "displayName-count-other": "Usbekistan-Sum",
        "symbol": "UZS"
    },
    "VEB": {
        "displayName": "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-one": "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-other": "Venezolanische Bolíva
2008)",
        "symbol": "VEB"
    },
    "VEF": {
        "displayName": "Venezolanischer Bolívar",
        "displayName-count-one": "Venezolanischer Bolívar",
        "displayName-count-other": "Venezolanische Bolíva
res",
        "symbol": "VEF",
        "symbol-alt-narrow": "Bs"
    },
    "VND": {
        "displayName": "Vietnamesischer Dong",
        "displayName-count-one": "Vietnamesischer Dong",
        "displayName-count-other": "Vietnamesische Dong",
        "symbol": "₫",
        "symbol-alt-narrow": "₫"
    },
    "VNN": {
        "displayName": "Vietnamesischer Dong (1978-1985)",
        "displayName-count-one": "Vietnamesischer Dong (1978-1985)",
        "displayName-count-other": "Vietnamesische Dong (1978-1985)",
        "symbol": "VNN"
    },
    "VUV": {
        "displayName": "Vanuatu-Vatu",
        "displayName-count-one": "Vanuatu-Vatu",
        "displayName-count-other": "Vanuatu-Vatu",
        "symbol": "VUV"
    },
    "WST": {

```

```

        "displayName": "Samoanischer Tala",
        "displayName-count-one": "Samoanischer Tala",
        "displayName-count-other": "Samoanische Tala",
        "symbol": "WST"
    },
    "XAF": {
        "displayName": "CFA-Franc (BEAC)",
        "displayName-count-one": "CFA-Franc (BEAC)",
        "displayName-count-other": "CFA-Franc (BEAC)",
        "symbol": "FCFA"
    },
    "XAG": {
        "displayName": "Unze Silber",
        "displayName-count-one": "Unze Silber",
        "displayName-count-other": "Unzen Silber",
        "symbol": "XAG"
    },
    "XAU": {
        "displayName": "Unze Gold",
        "displayName-count-one": "Unze Gold",
        "displayName-count-other": "Unzen Gold",
        "symbol": "XAU"
    },
    "XBA": {
        "displayName": "Europäische Rechnungseinheit",
        "displayName-count-one": "Europäische Rechnungseinheiten",
        "displayName-count-other": "Europäische Rechnungseinheiten",
        "symbol": "XBA"
    },
    "XBB": {
        "displayName": "Europäische Währungseinheit (XBB)",
        "displayName-count-one": "Europäische Währungseinheiten (XBB)",
        "displayName-count-other": "Europäische Währungseinheiten (XBB)",
        "symbol": "XBB"
    },
    "XBC": {
        "displayName": "Europäische Rechnungseinheit (XBC)",
        "displayName-count-one": "Europäische Rechnungseinheiten (XBC)",
        "displayName-count-other": "Europäische Rechnungseinheiten
(XBC) ",
        "symbol": "XBC"
    },
    "XBD": {
        "displayName": "Europäische Rechnungseinheit (XBD)",
        "displayName-count-one": "Europäische Rechnungseinheiten (XBD)",
        "displayName-count-other": "Europäische Rechnungseinheiten
(XBD) ",
        "symbol": "XBD"
    },
    "XCD": {
        "displayName": "Ostkaribischer Dollar",
        "displayName-count-one": "Ostkaribischer Dollar",
        "displayName-count-other": "Ostkaribische Dollar",
        "symbol": "EC$",
        "symbol-alt-narrow": "$"
    },
    "XDR": {

```



```

        "displayName": "Sonderziehungsrechte",
        "displayName-count-one": "Sonderziehungsrechte",
        "displayName-count-other": "Sonderziehungsrechte",
        "symbol": "XDR"
    },
    "XEU": {
        "displayName": "Europäische Währungseinheit (XEU)",
        "displayName-count-one": "Europäische Währungseinheiten (XEU)",
        "displayName-count-other": "Europäische Währungseinheiten (XEU)",
        "symbol": "XEU"
    },
    "XFO": {
        "displayName": "Französischer Gold-Franc",
        "displayName-count-one": "Französische Gold-Franc",
        "displayName-count-other": "Französische Gold-Franc",
        "symbol": "XFO"
    },
    "XFU": {
        "displayName": "Französischer UIC-Franc",
        "displayName-count-one": "Französische UIC-Franc",
        "displayName-count-other": "Französische UIC-Franc",
        "symbol": "XFU"
    },
    "XOF": {
        "displayName": "CFA-Franc (BCEAO)",
        "displayName-count-one": "CFA-Franc (BCEAO)",
        "displayName-count-other": "CFA-Francs (BCEAO)",
        "symbol": "CFA"
    },
    "XPD": {
        "displayName": "Unze Palladium",
        "displayName-count-one": "Unze Palladium",
        "displayName-count-other": "Unzen Palladium",
        "symbol": "XPD"
    },
    "XPF": {
        "displayName": "CFP-Franc",
        "displayName-count-one": "CFP-Franc",
        "displayName-count-other": "CFP-Franc",
        "symbol": "CFPF"
    },
    "XPT": {
        "displayName": "Unze Platin",
        "displayName-count-one": "Unze Platin",
        "displayName-count-other": "Unzen Platin",
        "symbol": "XPT"
    },
    "XRE": {
        "displayName": "RINET Funds",
        "displayName-count-one": "RINET Funds",
        "displayName-count-other": "RINET Funds",
        "symbol": "XRE"
    },
    "XSU": {
        "displayName": "SUCRE",
        "displayName-count-one": "SUCRE",
        "displayName-count-other": "SUCRE",

```

```

        "symbol": "XSU"
    },
    "XTS": {
        "displayName": "Testwährung",
        "displayName-count-one": "Testwährung",
        "displayName-count-other": "Testwährung",
        "symbol": "XTS"
    },
    "XUA": {
        "displayName": "Rechnungseinheit der AfEB",
        "displayName-count-one": "Rechnungseinheit der AfEB",
        "displayName-count-other": "Rechnungseinheiten der AfEB",
        "symbol": "XUA"
    },
    "XXX": {
        "displayName": "Unbekannte Währung",
        "displayName-count-one": "(unbekannte Währung)",
        "displayName-count-other": "(unbekannte Währung)",
        "symbol": "XXX"
    },
    "YDD": {
        "displayName": "Jemen-Dinar",
        "displayName-count-one": "Jemen-Dinar",
        "displayName-count-other": "Jemen-Dinar",
        "symbol": "YDD"
    },
    "YER": {
        "displayName": "Jemen-Rial",
        "displayName-count-one": "Jemen-Rial",
        "displayName-count-other": "Jemen-Rial",
        "symbol": "YER"
    },
    "YUD": {
        "displayName": "Jugoslawischer Dinar (1966-1990)",
        "displayName-count-one": "Jugoslawischer Dinar (1966-1990)",
        "displayName-count-other": "Jugoslawische Dinar (1966-1990)",
        "symbol": "YUD"
    },
    "YUM": {
        "displayName": "Jugoslawischer Neuer Dinar (1994-2002)",
        "displayName-count-one": "Jugoslawischer Neuer Dinar (1994-2002)",
        "displayName-count-other": "Jugoslawische Neue Dinar (1994-2002)",
        "symbol": "YUM"
    },
    "YUN": {
        "displayName": "Jugoslawischer Dinar (konvertibel)",
        "displayName-count-one": "Jugoslawische Dinar (konvertibel)",
        "displayName-count-other": "Jugoslawische Dinar (konvertibel)",
        "symbol": "YUN"
    },
    "YUR": {
        "displayName": "Jugoslawischer reformierter Dinar (1992-1993)",
        "displayName-count-one": "Jugoslawischer reformierter Dinar (1992-1993)",

```

```

        "displayName-count-other": "Jugoslawische reformierte Dinar
(1992-1993)",
        "symbol": "YUR"
    },
    "ZAL": {
        "displayName": "Südafrikanischer Rand (Finanz)",
        "displayName-count-one": "Südafrikanischer Rand (Finanz)",
        "displayName-count-other": "Südafrikanischer Rand (Finanz)",
        "symbol": "ZAL"
    },
    "ZAR": {
        "displayName": "Südafrikanischer Rand",
        "displayName-count-one": "Südafrikanischer Rand",
        "displayName-count-other": "Südafrikanische Rand",
        "symbol": "ZAR",
        "symbol-alt-narrow": "R"
    },
    "ZMK": {
        "displayName": "Kwacha (1968-2012)",
        "displayName-count-one": "Kwacha (1968-2012)",
        "displayName-count-other": "Kwacha (1968-2012)",
        "symbol": "ZMK"
    },
    "ZMW": {
        "displayName": "Kwacha",
        "displayName-count-one": "Kwacha",
        "displayName-count-other": "Kwacha",
        "symbol": "ZMW",
        "symbol-alt-narrow": "K"
    },
    "ZRN": {
        "displayName": "Zaire-Neuer Zaïre (1993-1998)",
        "displayName-count-one": "Zaire-Neuer Zaïre (1993-1998)",
        "displayName-count-other": "Zaire-Neue Zaïre (1993-1998)",
        "symbol": "ZRN"
    },
    "ZRZ": {
        "displayName": "Zaire-Zaïre (1971-1993)",
        "displayName-count-one": "Zaire-Zaïre (1971-1993)",
        "displayName-count-other": "Zaire-Zaïre (1971-1993)",
        "symbol": "ZRZ"
    },
    "ZWD": {
        "displayName": "Simbabwe-Dollar (1980-2008)",
        "displayName-count-one": "Simbabwe-Dollar (1980-2008)",
        "displayName-count-other": "Simbabwe-Dollar (1980-2008)",
        "symbol": "ZWD"
    },
    "ZWL": {
        "displayName": "Simbabwe-Dollar (2009)",
        "displayName-count-one": "Simbabwe-Dollar (2009)",
        "displayName-count-other": "Simbabwe-Dollar (2009)",
        "symbol": "ZWL"
    },
    "ZWR": {
        "displayName": "Simbabwe-Dollar (2008)",
        "displayName-count-one": "Simbabwe-Dollar (2008)",

```

```

        "displayName-count-other": "Simbabwe-Dollar (2008)",
        "symbol": "ZWR"
    }
}
}
}
}
}
}
}

```

CURRENCIES.JSX

```

{
    "main";
    {
        "de";
        {
            "identity";
            {
                "version";
                {
                    "_number";
                    "$Revision: 13259 $",
                    "_cldrVersion";
                    "31";
                }
                "language";
                "de";
            }
            "numbers";
            {
                "currencies";
                {
                    "ADP";
                    {
                        "displayName";
                        "Andorranische Pesete",
                        "displayName-count-one";
                        "Andorranische Pesete",
                        "displayName-count-other";
                        "Andorranische Peseten",
                        "symbol";
                        "ADP";
                    }
                    "AED";
                    {
                        "displayName";
                        "VAE-Dirham",
                        "displayName-count-one";
                        "VAE-Dirham",
                        "displayName-count-other";
                        "VAE-Dirham",
                        "symbol";
                        "AED";
                    }
                    "AFA";
                    {

```

```

        "displayName";
        "Afghanische Afghani (1927-2002)",
        "displayName-count-one";
        "Afghanische Afghani (1927-2002)",
        "displayName-count-other";
        "Afghanische Afghani (1927-2002)",
        "symbol";
        "AFA";
    }
    "AFN";
    {
        "displayName";
        "Afghanischer Afghani",
        "displayName-count-one";
        "Afghanischer Afghani",
        "displayName-count-other";
        "Afghanische Afghani",
        "symbol";
        "AFN";
    }
    "ALK";
    {
        "displayName";
        "Albanischer Lek (1946-1965)",
        "displayName-count-one";
        "Albanischer Lek (1946-1965)",
        "displayName-count-other";
        "Albanische Lek (1946-1965)";
    }
    "ALL";
    {
        "displayName";
        "Albanischer Lek",
        "displayName-count-one";
        "Albanischer Lek",
        "displayName-count-other";
        "Albanische Lek",
        "symbol";
        "ALL";
    }
    "AMD";
    {
        "displayName";
        "Armenischer Dram",
        "displayName-count-one";
        "Armenischer Dram",
        "displayName-count-other";
        "Armenische Dram",
        "symbol";
        "AMD";
    }
    "ANG";
    {
        "displayName";
        "Niederländische-Antillen-Gulden",
        "displayName-count-one";
        "Niederländische-Antillen-Gulden",

```

```

        "displayName-count-other";
        "Niederländische-Antillen-Gulden",
        "symbol";
        "ANG";
    }
    "AOA";
    {
        "displayName";
        "Angolanischer Kwanza",
        "displayName-count-one";
        "Angolanischer Kwanza",
        "displayName-count-other";
        "Angolanische Kwanza",
        "symbol";
        "AOA",
        "symbol-alt-narrow";
        "Kz";
    }
    "AOK";
    {
        "displayName";
        "Angolanischer Kwanza (1977-1990)",
        "displayName-count-one";
        "Angolanischer Kwanza (1977-1990)",
        "displayName-count-other";
        "Angolanische Kwanza (1977-1990)",
        "symbol";
        "AOK";
    }
    "AON";
    {
        "displayName";
        "Angolanischer Neuer Kwanza (1990-2000)",
        "displayName-count-one";
        "Angolanischer Neuer Kwanza (1990-2000)",
        "displayName-count-other";
        "Angolanische Neue Kwanza (1990-2000)",
        "symbol";
        "AON";
    }
    "AOR";
    {
        "displayName";
        "Angolanischer Kwanza Reajustado (1995-1999)",
        "displayName-count-one";
        "Angolanischer Kwanza Reajustado (1995-1999)",
        "displayName-count-other";
        "Angolanische Kwanza Reajustado (1995-1999)",
        "symbol";
        "AOR";
    }
    "ARA";
    {
        "displayName";
        "Argentinischer Austral",
        "displayName-count-one";
        "Argentinischer Austral",

```

```

        "displayName-count-other";
        "Argentinische Austral",
        "symbol";
        "ARA";
    }
    "ARL";
    {
        "displayName";
        "Argentinischer Peso Ley (1970-1983)",
        "displayName-count-one";
        "Argentinischer Peso Ley (1970-1983)",
        "displayName-count-other";
        "Argentinische Pesos Ley (1970-1983)",
        "symbol";
        "ARL";
    }
    "ARM";
    {
        "displayName";
        "Argentinischer Peso (1881-1970)",
        "displayName-count-one";
        "Argentinischer Peso (1881-1970)",
        "displayName-count-other";
        "Argentinische Pesos (1881-1970)",
        "symbol";
        "ARM";
    }
    "ARP";
    {
        "displayName";
        "Argentinischer Peso (1983-1985)",
        "displayName-count-one";
        "Argentinischer Peso (1983-1985)",
        "displayName-count-other";
        "Argentinische Peso (1983-1985)",
        "symbol";
        "ARP";
    }
    "ARS";
    {
        "displayName";
        "Argentinischer Peso",
        "displayName-count-one";
        "Argentinischer Peso",
        "displayName-count-other";
        "Argentinische Pesos",
        "symbol";
        "ARS",
        "symbol-alt-narrow";
        "$";
    }
    "ATS";
    {
        "displayName";
        "Österreichischer Schilling",
        "displayName-count-one";
        "Österreichischer Schilling",

```

```

        "displayName-count-other";
        "Österreichische Schilling",
        "symbol";
        "öS";
    }
    "AUD";
    {
        "displayName";
        "Australischer Dollar",
        "displayName-count-one";
        "Australischer Dollar",
        "displayName-count-other";
        "Australische Dollar",
        "symbol";
        "AU$";
        "symbol-alt-narrow";
        "$";
    }
    "AWG";
    {
        "displayName";
        "Aruba-Florin",
        "displayName-count-one";
        "Aruba-Florin",
        "displayName-count-other";
        "Aruba-Florin",
        "symbol";
        "AWG";
    }
    "AZM";
    {
        "displayName";
        "Aserbaidtschan-Manat (1993-2006)",
        "displayName-count-one";
        "Aserbaidtschan-Manat (1993-2006)",
        "displayName-count-other";
        "Aserbaidtschan-Manat (1993-2006)",
        "symbol";
        "AZM";
    }
    "AZN";
    {
        "displayName";
        "Aserbaidtschan-Manat",
        "displayName-count-one";
        "Aserbaidtschan-Manat",
        "displayName-count-other";
        "Aserbaidtschan-Manat",
        "symbol";
        "AZN";
    }
    "BAD";
    {
        "displayName";
        "Bosnien und Herzegowina Dinar (1992-1994)",
        "displayName-count-one";
        "Bosnien und Herzegowina Dinar (1992-1994)",

```



```

        "displayName-count-other";
        "Bosnien und Herzegowina Dinar (1992-1994)",
        "symbol";
        "BAD";
    }
    "BAM";
    {
        "displayName";
        "Bosnien und Herzegowina Konvertierbare Mark",
        "displayName-count-one";
        "Bosnien und Herzegowina Konvertierbare Mark",
        "displayName-count-other";
        "Bosnien und Herzegowina Konvertierbare Mark",
        "symbol";
        "BAM",
        "symbol-alt-narrow";
        "KM";
    }
    "BAN";
    {
        "displayName";
        "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
        "displayName-count-one";
        "Bosnien und Herzegowina Neuer Dinar (1994-1997)",
        "displayName-count-other";
        "Bosnien und Herzegowina Neue Dinar (1994-1997)",
        "symbol";
        "BAN";
    }
    "BBD";
    {
        "displayName";
        "Barbados-Dollar",
        "displayName-count-one";
        "Barbados-Dollar",
        "displayName-count-other";
        "Barbados-Dollar",
        "symbol";
        "BBD",
        "symbol-alt-narrow";
        "$";
    }
    "BDT";
    {
        "displayName";
        "Bangladesch-Taka",
        "displayName-count-one";
        "Bangladesch-Taka",
        "displayName-count-other";
        "Bangladesch-Taka",
        "symbol";
        "BDT",
        "symbol-alt-narrow";
        "৳";
    }
    "BEC";

```

```

    {
      "displayName";
      "Belgischer Franc (konvertibel)",
      "displayName-count-one";
      "Belgischer Franc (konvertibel)",
      "displayName-count-other";
      "Belgische Franc (konvertibel)",
      "symbol";
      "BEC";
    }
    "BEF";
    {
      "displayName";
      "Belgischer Franc",
      "displayName-count-one";
      "Belgischer Franc",
      "displayName-count-other";
      "Belgische Franc",
      "symbol";
      "BEF";
    }
    "BEL";
    {
      "displayName";
      "Belgischer Finanz-Franc",
      "displayName-count-one";
      "Belgischer Finanz-Franc",
      "displayName-count-other";
      "Belgische Finanz-Franc",
      "symbol";
      "BEL";
    }
    "BGL";
    {
      "displayName";
      "Bulgarische Lew (1962-1999)",
      "displayName-count-one";
      "Bulgarische Lew (1962-1999)",
      "displayName-count-other";
      "Bulgarische Lew (1962-1999)",
      "symbol";
      "BGL";
    }
    "BGM";
    {
      "displayName";
      "Bulgarischer Lew (1952-1962)",
      "displayName-count-one";
      "Bulgarischer Lew (1952-1962)",
      "displayName-count-other";
      "Bulgarische Lew (1952-1962)",
      "symbol";
      "BGK";
    }
    "BGN";
    {
      "displayName";

```

```

        "Bulgarischer Lew",
        "displayName-count-one";
        "Bulgarischer Lew",
        "displayName-count-other";
        "Bulgarische Lew",
        "symbol";
        "BGN";
    }
    "BGO";
    {
        "displayName";
        "Bulgarischer Lew (1879-1952)",
        "displayName-count-one";
        "Bulgarischer Lew (1879-1952)",
        "displayName-count-other";
        "Bulgarische Lew (1879-1952)",
        "symbol";
        "BGJ";
    }
    "BHD";
    {
        "displayName";
        "Bahrain-Dinar",
        "displayName-count-one";
        "Bahrain-Dinar",
        "displayName-count-other";
        "Bahrain-Dinar",
        "symbol";
        "BHD";
    }
    "BIF";
    {
        "displayName";
        "Burundi-Franc",
        "displayName-count-one";
        "Burundi-Franc",
        "displayName-count-other";
        "Burundi-Francs",
        "symbol";
        "BIF";
    }
    "BMD";
    {
        "displayName";
        "Bermuda-Dollar",
        "displayName-count-one";
        "Bermuda-Dollar",
        "displayName-count-other";
        "Bermuda-Dollar",
        "symbol";
        "BMD",
        "symbol-alt-narrow";
        "$";
    }
    "BND";
    {
        "displayName";

```

```

        "Brunei-Dollar",
        "displayName-count-one";
        "Brunei-Dollar",
        "displayName-count-other";
        "Brunei-Dollar",
        "symbol";
        "BND",
        "symbol-alt-narrow";
        "$";
    }
    "BOB";
    {
        "displayName";
        "Bolivanischer Boliviano",
        "displayName-count-one";
        "Bolivanischer Boliviano",
        "displayName-count-other";
        "Bolivianische Bolivianos",
        "symbol";
        "BOB",
        "symbol-alt-narrow";
        "Bs";
    }
    "BOL";
    {
        "displayName";
        "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-one";
        "Bolivianischer Boliviano (1863-1963)",
        "displayName-count-other";
        "Bolivianische Bolivianos (1863-1963)",
        "symbol";
        "BOL";
    }
    "BOP";
    {
        "displayName";
        "Bolivianischer Peso",
        "displayName-count-one";
        "Bolivianischer Peso",
        "displayName-count-other";
        "Bolivianische Peso",
        "symbol";
        "BOP";
    }
    "BOV";
    {
        "displayName";
        "Boliviansiche Mvdol",
        "displayName-count-one";
        "Boliviansiche Mvdol",
        "displayName-count-other";
        "Bolivianische Mvdol",
        "symbol";
        "BOV";
    }
    "BRB";

```

```

    {
      "displayName";
      "Brasilianischer Cruzeiro Novo (1967-1986)",
      "displayName-count-one";
      "Brasilianischer Cruzeiro Novo (1967-1986)",
      "displayName-count-other";
      "Brasilianische Cruzeiro Novo (1967-1986)",
      "symbol";
      "BRB";
    }
    "BRC";
    {
      "displayName";
      "Brasilianischer Cruzado (1986-1989)",
      "displayName-count-one";
      "Brasilianischer Cruzado (1986-1989)",
      "displayName-count-other";
      "Brasilianische Cruzado (1986-1989)",
      "symbol";
      "BRC";
    }
    "BRE";
    {
      "displayName";
      "Brasilianischer Cruzeiro (1990-1993)",
      "displayName-count-one";
      "Brasilianischer Cruzeiro (1990-1993)",
      "displayName-count-other";
      "Brasilianische Cruzeiro (1990-1993)",
      "symbol";
      "BRE";
    }
    "BRL";
    {
      "displayName";
      "Brasilianischer Real",
      "displayName-count-one";
      "Brasilianischer Real",
      "displayName-count-other";
      "Brasilianische Real",
      "symbol";
      "R$";
      "symbol-alt-narrow";
      "R$";
    }
    "BRN";
    {
      "displayName";
      "Brasilianischer Cruzado Novo (1989-1990)",
      "displayName-count-one";
      "Brasilianischer Cruzado Novo (1989-1990)",
      "displayName-count-other";
      "Brasilianische Cruzado Novo (1989-1990)",
      "symbol";
      "BRN";
    }
    "BRR";

```

```

    {
      "displayName";
      "Brasilianischer Cruzeiro (1993-1994)",
      "displayName-count-one";
      "Brasilianischer Cruzeiro (1993-1994)",
      "displayName-count-other";
      "Brasilianische Cruzeiro (1993-1994)",
      "symbol";
      "BRR";
    }
    "BRZ";
    {
      "displayName";
      "Brasilianischer Cruzeiro (1942-1967)",
      "displayName-count-one";
      "Brasilianischer Cruzeiro (1942-1967)",
      "displayName-count-other";
      "Brasilianischer Cruzeiro (1942-1967)",
      "symbol";
      "BRZ";
    }
    "BSD";
    {
      "displayName";
      "Bahamas-Dollar",
      "displayName-count-one";
      "Bahamas-Dollar",
      "displayName-count-other";
      "Bahamas-Dollar",
      "symbol";
      "BSD",
      "symbol-alt-narrow";
      "$";
    }
    "BTN";
    {
      "displayName";
      "Bhutan-Ngultrum",
      "displayName-count-one";
      "Bhutan-Ngultrum",
      "displayName-count-other";
      "Bhutan-Ngultrum",
      "symbol";
      "BTN";
    }
    "BUK";
    {
      "displayName";
      "Birmanischer Kyat",
      "displayName-count-one";
      "Birmanischer Kyat",
      "displayName-count-other";
      "Birmanische Kyat",
      "symbol";
      "BUK";
    }
    "BWP";

```

```

    {
      "displayName";
      "Botswanischer Pula",
      "displayName-count-one";
      "Botswanischer Pula",
      "displayName-count-other";
      "Botswanische Pula",
      "symbol";
      "BWP",
      "symbol-alt-narrow";
      "P";
    }
    "BYB";
    {
      "displayName";
      "Belarus-Rubel (1994-1999)",
      "displayName-count-one";
      "Belarus-Rubel (1994-1999)",
      "displayName-count-other";
      "Belarus-Rubel (1994-1999)",
      "symbol";
      "BYB";
    }
    "BYN";
    {
      "displayName";
      "Weißrussischer Rubel",
      "displayName-count-one";
      "Weißrussischer Rubel",
      "displayName-count-other";
      "Weißrussische Rubel",
      "symbol";
      "BYN",
      "symbol-alt-narrow";
      "p.";
    }
    "BYR";
    {
      "displayName";
      "Weißrussischer Rubel (2000-2016)",
      "displayName-count-one";
      "Weißrussischer Rubel (2000-2016)",
      "displayName-count-other";
      "Weißrussische Rubel (2000-2016)",
      "symbol";
      "BYR";
    }
    "BZD";
    {
      "displayName";
      "Belize-Dollar",
      "displayName-count-one";
      "Belize-Dollar",
      "displayName-count-other";
      "Belize-Dollar",
      "symbol";
      "BZD",

```

```

        "symbol-alt-narrow";
        "$";
    }
    "CAD";
    {
        "displayName";
        "Kanadischer Dollar",
        "displayName-count-one";
        "Kanadischer Dollar",
        "displayName-count-other";
        "Kanadische Dollar",
        "symbol";
        "CA$",
        "symbol-alt-narrow";
        "$";
    }
    "CDF";
    {
        "displayName";
        "Kongo-Franc",
        "displayName-count-one";
        "Kongo-Franc",
        "displayName-count-other";
        "Kongo-Francs",
        "symbol";
        "CDF";
    }
    "CHE";
    {
        "displayName";
        "WIR-Euro",
        "displayName-count-one";
        "WIR-Euro",
        "displayName-count-other";
        "WIR-Euro",
        "symbol";
        "CHE";
    }
    "CHF";
    {
        "displayName";
        "Schweizer Franken",
        "displayName-count-one";
        "Schweizer Franken",
        "displayName-count-other";
        "Schweizer Franken",
        "symbol";
        "CHF";
    }
    "CHW";
    {
        "displayName";
        "WIR Franken",
        "displayName-count-one";
        "WIR Franken",
        "displayName-count-other";
        "WIR Franken",

```



```

        "symbol";
        "CHW";
    }
    "CLE";
    {
        "displayName";
        "Chilenischer Escudo",
        "displayName-count-one";
        "Chilenischer Escudo",
        "displayName-count-other";
        "Chilenische Escudo",
        "symbol";
        "CLE";
    }
    "CLF";
    {
        "displayName";
        "Chilenische Unidades de Fomento",
        "displayName-count-one";
        "Chilenische Unidades de Fomento",
        "displayName-count-other";
        "Chilenische Unidades de Fomento",
        "symbol";
        "CLF";
    }
    "CLP";
    {
        "displayName";
        "Chilenischer Peso",
        "displayName-count-one";
        "Chilenischer Peso",
        "displayName-count-other";
        "Chilenische Pesos",
        "symbol";
        "CLP",
        "symbol-alt-narrow";
        "$";
    }
    "CNX";
    {
        "displayName";
        "Dollar der Chinesischen Volksbank",
        "displayName-count-one";
        "Dollar der Chinesischen Volksbank",
        "displayName-count-other";
        "Dollar der Chinesischen Volksbank",
        "symbol";
        "CNX";
    }
    "CNY";
    {
        "displayName";
        "Renminbi Yuan",
        "displayName-count-one";
        "Chinesischer Yuan",
        "displayName-count-other";
        "Renminbi Yuan",

```

```

        "symbol";
        "CN¥",
        "symbol-alt-narrow";
        "¥";
    }
    "COP";
    {
        "displayName";
        "Kolumbianischer Peso",
        "displayName-count-one";
        "Kolumbianischer Peso",
        "displayName-count-other";
        "Kolumbianische Pesos",
        "symbol";
        "COP",
        "symbol-alt-narrow";
        "$";
    }
    "COU";
    {
        "displayName";
        "Kolumbianische Unidades de valor real",
        "displayName-count-one";
        "Kolumbianische Unidad de valor real",
        "displayName-count-other";
        "Kolumbianische Unidades de valor real",
        "symbol";
        "COU";
    }
    "CRC";
    {
        "displayName";
        "Costa-Rica-Colón",
        "displayName-count-one";
        "Costa-Rica-Colón",
        "displayName-count-other";
        "Costa-Rica-Colón",
        "symbol";
        "CRC",
        "symbol-alt-narrow";
        "₡";
    }
    "CSD";
    {
        "displayName";
        "Serbischer Dinar (2002-2006)",
        "displayName-count-one";
        "Serbischer Dinar (2002-2006)",
        "displayName-count-other";
        "Serbische Dinar (2002-2006)",
        "symbol";
        "CSD";
    }
    "CSK";
    {
        "displayName";
        "Tschechoslowakische Krone",

```

```

        "displayName-count-one";
        "Tschechoslowakische Kronen",
        "displayName-count-other";
        "Tschechoslowakische Kronen",
        "symbol";
        "CSK";
    }
    "CUC";
    {
        "displayName";
        "Kubanischer Peso (konvertibel)",
        "displayName-count-one";
        "Kubanischer Peso (konvertibel)",
        "displayName-count-other";
        "Kubanische Pesos (konvertibel)",
        "symbol";
        "CUC",
        "symbol-alt-narrow";
        "Cub$";
    }
    "CUP";
    {
        "displayName";
        "Kubanischer Peso",
        "displayName-count-one";
        "Kubanischer Peso",
        "displayName-count-other";
        "Kubanische Pesos",
        "symbol";
        "CUP",
        "symbol-alt-narrow";
        "$";
    }
    "CVE";
    {
        "displayName";
        "Cabo-Verde-Escudo",
        "displayName-count-one";
        "Cabo-Verde-Escudo",
        "displayName-count-other";
        "Cabo-Verde-Escudos",
        "symbol";
        "CVE";
    }
    "CYP";
    {
        "displayName";
        "Zypern-Pfund",
        "displayName-count-one";
        "Zypern Pfund",
        "displayName-count-other";
        "Zypern Pfund",
        "symbol";
        "CYP";
    }
    "CZK";
    {

```

```

        "displayName";
        "Tschechische Krone",
        "displayName-count-one";
        "Tschechische Krone",
        "displayName-count-other";
        "Tschechische Kronen",
        "symbol";
        "CZK",
        "symbol-alt-narrow";
        "Kč";
    }
    "DDM";
    {
        "displayName";
        "Mark der DDR",
        "displayName-count-one";
        "Mark der DDR",
        "displayName-count-other";
        "Mark der DDR",
        "symbol";
        "DDM";
    }
    "DEM";
    {
        "displayName";
        "Deutsche Mark",
        "displayName-count-one";
        "Deutsche Mark",
        "displayName-count-other";
        "Deutsche Mark",
        "symbol";
        "DM";
    }
    "DJF";
    {
        "displayName";
        "Dschibuti-Franc",
        "displayName-count-one";
        "Dschibuti-Franc",
        "displayName-count-other";
        "Dschibuti-Franc",
        "symbol";
        "DJF";
    }
    "DKK";
    {
        "displayName";
        "Dänische Krone",
        "displayName-count-one";
        "Dänische Krone",
        "displayName-count-other";
        "Dänische Kronen",
        "symbol";
        "DKK",
        "symbol-alt-narrow";
        "kr";
    }
}

```

```
"DOP";
{
  "displayName";
  "Dominikanischer Peso",
    "displayName-count-one";
  "Dominikanischer Peso",
    "displayName-count-other";
  "Dominikanische Pesos",
    "symbol";
  "DOP",
    "symbol-alt-narrow";
  "$";
}
"DZD";
{
  "displayName";
  "Algerischer Dinar",
    "displayName-count-one";
  "Algerischer Dinar",
    "displayName-count-other";
  "Algerische Dinar",
    "symbol";
  "DZD";
}
"ECS";
{
  "displayName";
  "Ecuadorianischer Sucre",
    "displayName-count-one";
  "Ecuadorianischer Sucre",
    "displayName-count-other";
  "Ecuadorianische Sucre",
    "symbol";
  "ECS";
}
"ECV";
{
  "displayName";
  "Verrechnungseinheit für Ecuador",
    "displayName-count-one";
  "Verrechnungseinheiten für Ecuador",
    "displayName-count-other";
  "Verrechnungseinheiten für Ecuador",
    "symbol";
  "ECV";
}
"EEK";
{
  "displayName";
  "Estnische Krone",
    "displayName-count-one";
  "Estnische Krone",
    "displayName-count-other";
  "Estnische Kronen",
    "symbol";
  "EEK";
}
```

```

"EGP";
{
  "displayName";
  "Ägyptisches Pfund",
    "displayName-count-one";
  "Ägyptisches Pfund",
    "displayName-count-other";
  "Ägyptische Pfund",
    "symbol";
  "EGP",
    "symbol-alt-narrow";
  "£";
}
"ERN";
{
  "displayName";
  "Eritreischer Nakfa",
    "displayName-count-one";
  "Eritreischer Nakfa",
    "displayName-count-other";
  "Eritreische Nakfa",
    "symbol";
  "ERN";
}
"ESA";
{
  "displayName";
  "Spanische Peseta (A-Konten)",
    "displayName-count-one";
  "Spanische Peseta (A-Konten)",
    "displayName-count-other";
  "Spanische Peseten (A-Konten)",
    "symbol";
  "ESA";
}
"ESB";
{
  "displayName";
  "Spanische Peseta (konvertibel)",
    "displayName-count-one";
  "Spanische Peseta (konvertibel)",
    "displayName-count-other";
  "Spanische Peseten (konvertibel)",
    "symbol";
  "ESB";
}
"ESP";
{
  "displayName";
  "Spanische Peseta",
    "displayName-count-one";
  "Spanische Peseta",
    "displayName-count-other";
  "Spanische Peseten",
    "symbol";
  "ESP",
    "symbol-alt-narrow";
}

```

```

        "E";
    }
    "ETB";
    {
        "displayName";
        "Äthiopischer Birr",
        "displayName-count-one";
        "Äthiopischer Birr",
        "displayName-count-other";
        "Äthiopische Birr",
        "symbol";
        "ETB";
    }
    "EUR";
    {
        "displayName";
        "Euro",
        "displayName-count-one";
        "Euro",
        "displayName-count-other";
        "Euro",
        "symbol";
        "€",
        "symbol-alt-narrow";
        "€";
    }
    "FIM";
    {
        "displayName";
        "Finnische Mark",
        "displayName-count-one";
        "Finnische Mark",
        "displayName-count-other";
        "Finnische Mark",
        "symbol";
        "FIM";
    }
    "FJD";
    {
        "displayName";
        "Fidschi-Dollar",
        "displayName-count-one";
        "Fidschi-Dollar",
        "displayName-count-other";
        "Fidschi-Dollar",
        "symbol";
        "FJD",
        "symbol-alt-narrow";
        "$";
    }
    "FKP";
    {
        "displayName";
        "Falkland-Pfund",
        "displayName-count-one";
        "Falkland-Pfund",
        "displayName-count-other";
    }

```

```

        "Falkland-Pfund",
        "symbol";
        "FKP",
        "symbol-alt-narrow";
        "Fl£";
    }
    "FRF";
    {
        "displayName";
        "Französischer Franc",
        "displayName-count-one";
        "Französischer Franc",
        "displayName-count-other";
        "Französische Franc",
        "symbol";
        "FRF";
    }
    "GBP";
    {
        "displayName";
        "Britisches Pfund",
        "displayName-count-one";
        "Britisches Pfund",
        "displayName-count-other";
        "Britische Pfund",
        "symbol";
        "£",
        "symbol-alt-narrow";
        "£";
    }
    "GEK";
    {
        "displayName";
        "Georgischer Kupon Larit",
        "displayName-count-one";
        "Georgischer Kupon Larit",
        "displayName-count-other";
        "Georgische Kupon Larit",
        "symbol";
        "GEK";
    }
    "GEL";
    {
        "displayName";
        "Georgischer Lari",
        "displayName-count-one";
        "Georgischer Lari",
        "displayName-count-other";
        "Georgische Lari",
        "symbol";
        "GEL",
        "symbol-alt-narrow";
        "ლ",
        "symbol-alt-variant";
        "ლ";
    }
    "GHC";

```



```

    {
      "displayName";
      "Ghanaischer Cedi (1979-2007)",
      "displayName-count-one";
      "Ghanaischer Cedi (1979-2007)",
      "displayName-count-other";
      "Ghanaische Cedi (1979-2007)",
      "symbol";
      "GHC";
    }
    "GHS";
    {
      "displayName";
      "Ghanaischer Cedi",
      "displayName-count-one";
      "Ghanaischer Cedi",
      "displayName-count-other";
      "Ghanaische Cedi",
      "symbol";
      "GHS";
    }
    "GIP";
    {
      "displayName";
      "Gibraltar-Pfund",
      "displayName-count-one";
      "Gibraltar-Pfund",
      "displayName-count-other";
      "Gibraltar Pfund",
      "symbol";
      "GIP",
      "symbol-alt-narrow";
      "£";
    }
    "GMD";
    {
      "displayName";
      "Gambia-Dalasi",
      "displayName-count-one";
      "Gambia-Dalasi",
      "displayName-count-other";
      "Gambia-Dalasi",
      "symbol";
      "GMD";
    }
    "GNF";
    {
      "displayName";
      "Guinea-Franc",
      "displayName-count-one";
      "Guinea-Franc",
      "displayName-count-other";
      "Guinea-Franc",
      "symbol";
      "GNF",
      "symbol-alt-narrow";
      "F.G.";
    }

```

```

    }
    "GNS";
    {
        "displayName";
        "Guineischer Syli",
        "displayName-count-one";
        "Guineischer Syli",
        "displayName-count-other";
        "Guineische Syli",
        "symbol";
        "GNS";
    }
    "GQE";
    {
        "displayName";
        "Äquatorialguinea-Ekwele",
        "displayName-count-one";
        "Äquatorialguinea-Ekwele",
        "displayName-count-other";
        "Äquatorialguinea-Ekwele",
        "symbol";
        "GQE";
    }
    "GRD";
    {
        "displayName";
        "Griechische Drachme",
        "displayName-count-one";
        "Griechische Drachme",
        "displayName-count-other";
        "Griechische Drachmen",
        "symbol";
        "GRD";
    }
    "GTQ";
    {
        "displayName";
        "Guatemaltekinscher Quetzal",
        "displayName-count-one";
        "Guatemaltekinscher Quetzal",
        "displayName-count-other";
        "Guatemaltekinsche Quetzales",
        "symbol";
        "GTQ",
        "symbol-alt-narrow";
        "Q";
    }
    "GWE";
    {
        "displayName";
        "Portugiesisch Guinea Escudo",
        "displayName-count-one";
        "Portugiesisch Guinea Escudo",
        "displayName-count-other";
        "Portugiesisch Guinea Escudo",
        "symbol";
        "GWE";
    }

```

```

    }
    "GWP";
    {
        "displayName";
        "Guinea-Bissau Peso",
        "displayName-count-one";
        "Guinea-Bissau Peso",
        "displayName-count-other";
        "Guinea-Bissau Pesos",
        "symbol";
        "GWP";
    }
    "GYD";
    {
        "displayName";
        "Guyana-Dollar",
        "displayName-count-one";
        "Guyana-Dollar",
        "displayName-count-other";
        "Guyana-Dollar",
        "symbol";
        "GYD",
        "symbol-alt-narrow";
        "$";
    }
    "HKD";
    {
        "displayName";
        "Hongkong-Dollar",
        "displayName-count-one";
        "Hongkong-Dollar",
        "displayName-count-other";
        "Hongkong-Dollar",
        "symbol";
        "HK$",
        "symbol-alt-narrow";
        "$";
    }
    "HNL";
    {
        "displayName";
        "Honduras-Lempira",
        "displayName-count-one";
        "Honduras-Lempira",
        "displayName-count-other";
        "Honduras-Lempira",
        "symbol";
        "HNL",
        "symbol-alt-narrow";
        "L";
    }
    "HRD";
    {
        "displayName";
        "Kroatischer Dinar",
        "displayName-count-one";
        "Kroatischer Dinar",

```

```

        "displayName-count-other";
        "Kroatische Dinar",
        "symbol";
        "HRD";
    }
    "HRK";
    {
        "displayName";
        "Kroatischer Kuna",
        "displayName-count-one";
        "Kroatischer Kuna",
        "displayName-count-other";
        "Kroatische Kuna",
        "symbol";
        "HRK",
        "symbol-alt-narrow";
        "kn";
    }
    "HTG";
    {
        "displayName";
        "Haitianische Gourde",
        "displayName-count-one";
        "Haitianische Gourde",
        "displayName-count-other";
        "Haitianische Gourdes",
        "symbol";
        "HTG";
    }
    "HUF";
    {
        "displayName";
        "Ungarischer Forint",
        "displayName-count-one";
        "Ungarischer Forint",
        "displayName-count-other";
        "Ungarische Forint",
        "symbol";
        "HUF",
        "symbol-alt-narrow";
        "Ft";
    }
    "IDR";
    {
        "displayName";
        "Indonesische Rupiah",
        "displayName-count-one";
        "Indonesische Rupiah",
        "displayName-count-other";
        "Indonesische Rupiah",
        "symbol";
        "IDR",
        "symbol-alt-narrow";
        "Rp";
    }
    "IEP";
    {

```

```

        "displayName";
        "Irisches Pfund",
        "displayName-count-one";
        "Irisches Pfund",
        "displayName-count-other";
        "Irische Pfund",
        "symbol";
        "IEP";
    }
    "ILP";
    {
        "displayName";
        "Israelisches Pfund",
        "displayName-count-one";
        "Israelisches Pfund",
        "displayName-count-other";
        "Israelische Pfund",
        "symbol";
        "ILP";
    }
    "ILR";
    {
        "displayName";
        "Israelischer Schekel (1980-1985)",
        "displayName-count-one";
        "Israelischer Schekel (1980-1985)",
        "displayName-count-other";
        "Israelische Schekel (1980-1985)";
    }
    "ILS";
    {
        "displayName";
        "Israelischer Neuer Schekel",
        "displayName-count-one";
        "Israelischer Neuer Schekel",
        "displayName-count-other";
        "Israelische Neue Schekel",
        "symbol";
        "₪",
        "symbol-alt-narrow";
        "₪";
    }
    "INR";
    {
        "displayName";
        "Indische Rupie",
        "displayName-count-one";
        "Indische Rupie",
        "displayName-count-other";
        "Indische Rupien",
        "symbol";
        "₹",
        "symbol-alt-narrow";
        "₹";
    }
    "IQD";
    {

```

```

        "displayName";
        "Irakischer Dinar",
        "displayName-count-one";
        "Irakischer Dinar",
        "displayName-count-other";
        "Irakische Dinar",
        "symbol";
        "IQD";
    }
    "IRR";
    {
        "displayName";
        "Iranischer Rial",
        "displayName-count-one";
        "Iranischer Rial",
        "displayName-count-other";
        "Iranische Rial",
        "symbol";
        "IRR";
    }
    "ISJ";
    {
        "displayName";
        "Isländische Krone (1918-1981)",
        "displayName-count-one";
        "Isländische Krone (1918-1981)",
        "displayName-count-other";
        "Isländische Kronen (1918-1981)";
    }
    "ISK";
    {
        "displayName";
        "Isländische Krone",
        "displayName-count-one";
        "Isländische Krone",
        "displayName-count-other";
        "Isländische Kronen",
        "symbol";
        "ISK",
        "symbol-alt-narrow";
        "kr";
    }
    "ITL";
    {
        "displayName";
        "Italienische Lira",
        "displayName-count-one";
        "Italienische Lira",
        "displayName-count-other";
        "Italienische Lire",
        "symbol";
        "ITL";
    }
    "JMD";
    {
        "displayName";
        "Jamaika-Dollar",

```

```

        "displayName-count-one";
        "Jamaika-Dollar",
        "displayName-count-other";
        "Jamaika-Dollar",
        "symbol";
        "JMD",
        "symbol-alt-narrow";
        "$";
    }
    "JOD";
    {
        "displayName";
        "Jordanischer Dinar",
        "displayName-count-one";
        "Jordanischer Dinar",
        "displayName-count-other";
        "Jordanische Dinar",
        "symbol";
        "JOD";
    }
    "JPY";
    {
        "displayName";
        "Japanischer Yen",
        "displayName-count-one";
        "Japanischer Yen",
        "displayName-count-other";
        "Japanische Yen",
        "symbol";
        "¥",
        "symbol-alt-narrow";
        "¥";
    }
    "KES";
    {
        "displayName";
        "Kenia-Schilling",
        "displayName-count-one";
        "Kenia-Schilling",
        "displayName-count-other";
        "Kenia-Schilling",
        "symbol";
        "KES";
    }
    "KGS";
    {
        "displayName";
        "Kirgisischer Som",
        "displayName-count-one";
        "Kirgisischer Som",
        "displayName-count-other";
        "Kirgisische Som",
        "symbol";
        "KGS";
    }
    "KHR";
    {

```

```

        "displayName";
        "Kambodschanischer Riel",
        "displayName-count-one";
        "Kambodschanischer Riel",
        "displayName-count-other";
        "Kambodschanische Riel",
        "symbol";
        "KHR",
        "symbol-alt-narrow";
        "៛";
    }
    "KMF";
    {
        "displayName";
        "Komoren-Franc",
        "displayName-count-one";
        "Komoren-Franc",
        "displayName-count-other";
        "Komoren-Francs",
        "symbol";
        "KMF",
        "symbol-alt-narrow";
        "FC";
    }
    "KPW";
    {
        "displayName";
        "Nordkoreanischer Won",
        "displayName-count-one";
        "Nordkoreanischer Won",
        "displayName-count-other";
        "Nordkoreanische Won",
        "symbol";
        "KPW",
        "symbol-alt-narrow";
        "₩";
    }
    "KRH";
    {
        "displayName";
        "Südkoreanischer Hwan (1953-1962)",
        "displayName-count-one";
        "Südkoreanischer Hwan (1953-1962)",
        "displayName-count-other";
        "Südkoreanischer Hwan (1953-1962)",
        "symbol";
        "KRH";
    }
    "KRO";
    {
        "displayName";
        "Südkoreanischer Won (1945-1953)",
        "displayName-count-one";
        "Südkoreanischer Won (1945-1953)",
        "displayName-count-other";
        "Südkoreanischer Won (1945-1953)",

```



```

        "symbol";
        "KRO";
    }
    "KRW";
    {
        "displayName";
        "Südkoreanischer Won",
        "displayName-count-one";
        "Südkoreanischer Won",
        "displayName-count-other";
        "Südkoreanische Won",
        "symbol";
        "₩",
        "symbol-alt-narrow";
        "₩";
    }
    "KWD";
    {
        "displayName";
        "Kuwait-Dinar",
        "displayName-count-one";
        "Kuwait-Dinar",
        "displayName-count-other";
        "Kuwait-Dinar",
        "symbol";
        "KWD";
    }
    "KYD";
    {
        "displayName";
        "Kaiman-Dollar",
        "displayName-count-one";
        "Kaiman-Dollar",
        "displayName-count-other";
        "Kaiman-Dollar",
        "symbol";
        "KYD",
        "symbol-alt-narrow";
        "$";
    }
    "KZT";
    {
        "displayName";
        "Kasachischer Tenge",
        "displayName-count-one";
        "Kasachischer Tenge",
        "displayName-count-other";
        "Kasachische Tenge",
        "symbol";
        "KZT",
        "symbol-alt-narrow";
        "₸";
    }
    "LAK";
    {
        "displayName";
        "Laotischer Kip",

```

```

        "displayName-count-one";
        "Laotischer Kip",
        "displayName-count-other";
        "Laotische Kip",
        "symbol";
        "LAK",
        "symbol-alt-narrow";
        "₭";
    }
    "LBP";
    {
        "displayName";
        "Libanesisches Pfund",
        "displayName-count-one";
        "Libanesisches Pfund",
        "displayName-count-other";
        "Libanesische Pfund",
        "symbol";
        "LBP",
        "symbol-alt-narrow";
        "L£";
    }
    "LKR";
    {
        "displayName";
        "Sri-Lanka-Rupie",
        "displayName-count-one";
        "Sri-Lanka-Rupie",
        "displayName-count-other";
        "Sri-Lanka-Rupien",
        "symbol";
        "LKR",
        "symbol-alt-narrow";
        "Rs";
    }
    "LRD";
    {
        "displayName";
        "Liberianischer Dollar",
        "displayName-count-one";
        "Liberianischer Dollar",
        "displayName-count-other";
        "Liberianische Dollar",
        "symbol";
        "LRD",
        "symbol-alt-narrow";
        "$";
    }
    "LSL";
    {
        "displayName";
        "Loti",
        "displayName-count-one";
        "Loti",
        "displayName-count-other";
        "Loti",
        "symbol";
    }

```

```

        "LSL";
    }
    "LTL";
    {
        "displayName";
        "Litauischer Litas",
        "displayName-count-one";
        "Litauischer Litas",
        "displayName-count-other";
        "Litauische Litas",
        "symbol";
        "LTL",
        "symbol-alt-narrow";
        "Lt";
    }
    "LTT";
    {
        "displayName";
        "Litauischer Talonas",
        "displayName-count-one";
        "Litauische Talonas",
        "displayName-count-other";
        "Litauische Talonas",
        "symbol";
        "LTT";
    }
    "LUC";
    {
        "displayName";
        "Luxemburgischer Franc (konvertibel)",
        "displayName-count-one";
        "Luxemburgische Franc (konvertibel)",
        "displayName-count-other";
        "Luxemburgische Franc (konvertibel)",
        "symbol";
        "LUC";
    }
    "LUF";
    {
        "displayName";
        "Luxemburgischer Franc",
        "displayName-count-one";
        "Luxemburgische Franc",
        "displayName-count-other";
        "Luxemburgische Franc",
        "symbol";
        "LUF";
    }
    "LUL";
    {
        "displayName";
        "Luxemburgischer Finanz-Franc",
        "displayName-count-one";
        "Luxemburgische Finanz-Franc",
        "displayName-count-other";
        "Luxemburgische Finanz-Franc",
        "symbol";
    }

```

```

        "LUL";
    }
    "LVL";
    {
        "displayName";
        "Lettischer Lats",
            "displayName-count-one";
        "Lettischer Lats",
            "displayName-count-other";
        "Lettische Lats",
            "symbol";
        "LVL",
            "symbol-alt-narrow";
        "Ls";
    }
    "LVR";
    {
        "displayName";
        "Lettischer Rubel",
            "displayName-count-one";
        "Lettische Rubel",
            "displayName-count-other";
        "Lettische Rubel",
            "symbol";
        "LVR";
    }
    "LYD";
    {
        "displayName";
        "Libyscher Dinar",
            "displayName-count-one";
        "Libyscher Dinar",
            "displayName-count-other";
        "Libysche Dinar",
            "symbol";
        "LYD";
    }
    "MAD";
    {
        "displayName";
        "Marokkanischer Dirham",
            "displayName-count-one";
        "Marokkanischer Dirham",
            "displayName-count-other";
        "Marokkanische Dirham",
            "symbol";
        "MAD";
    }
    "MAF";
    {
        "displayName";
        "Marokkanischer Franc",
            "displayName-count-one";
        "Marokkanische Franc",
            "displayName-count-other";
        "Marokkanische Franc",
            "symbol";
    }

```

```

        "MAF";
    }
    "MCF";
    {
        "displayName";
        "Monegassischer Franc",
        "displayName-count-one";
        "Monegassischer Franc",
        "displayName-count-other";
        "Monegassische Franc",
        "symbol";
        "MCF";
    }
    "MDC";
    {
        "displayName";
        "Moldau-Cupon",
        "displayName-count-one";
        "Moldau-Cupon",
        "displayName-count-other";
        "Moldau-Cupon",
        "symbol";
        "MDC";
    }
    "MDL";
    {
        "displayName";
        "Moldau-Leu",
        "displayName-count-one";
        "Moldau-Leu",
        "displayName-count-other";
        "Moldau-Leu",
        "symbol";
        "MDL";
    }
    "MGA";
    {
        "displayName";
        "Madagaskar-Ariary",
        "displayName-count-one";
        "Madagaskar-Ariary",
        "displayName-count-other";
        "Madagaskar-Ariary",
        "symbol";
        "MGA",
        "symbol-alt-narrow";
        "Ar";
    }
    "MGF";
    {
        "displayName";
        "Madagaskar-Franc",
        "displayName-count-one";
        "Madagaskar-Franc",
        "displayName-count-other";
        "Madagaskar-Franc",
        "symbol";
    }

```

```

        "MGF";
    }
    "MKD";
    {
        "displayName";
        "Mazedonischer Denar",
        "displayName-count-one";
        "Mazedonischer Denar",
        "displayName-count-other";
        "Mazedonische Denari",
        "symbol";
        "MKD";
    }
    "MKN";
    {
        "displayName";
        "Mazedonischer Denar (1992-1993)",
        "displayName-count-one";
        "Mazedonischer Denar (1992-1993)",
        "displayName-count-other";
        "Mazedonische Denar (1992-1993)",
        "symbol";
        "MKN";
    }
    "MLF";
    {
        "displayName";
        "Malischer Franc",
        "displayName-count-one";
        "Malische Franc",
        "displayName-count-other";
        "Malische Franc",
        "symbol";
        "MLF";
    }
    "MMK";
    {
        "displayName";
        "Myanmarischer Kyat",
        "displayName-count-one";
        "Myanmarischer Kyat",
        "displayName-count-other";
        "Myanmarische Kyat",
        "symbol";
        "MMK",
        "symbol-alt-narrow";
        "K";
    }
    "MNT";
    {
        "displayName";
        "Mongolischer Tögrög",
        "displayName-count-one";
        "Mongolischer Tögrög",
        "displayName-count-other";
        "Mongolische Tögrög",
        "symbol";
    }

```

```

        "MNT",
        "symbol-alt-narrow";
        "₭";
    }
    "MOP";
    {
        "displayName";
        "Macao-Pataca",
        "displayName-count-one";
        "Macao-Pataca",
        "displayName-count-other";
        "Macao-Pataca",
        "symbol";
        "MOP";
    }
    "MRO";
    {
        "displayName";
        "Mauretanischer Ouguiya",
        "displayName-count-one";
        "Mauretanischer Ouguiya",
        "displayName-count-other";
        "Mauretanische Ouguiya",
        "symbol";
        "MRO";
    }
    "MTL";
    {
        "displayName";
        "Maltesische Lira",
        "displayName-count-one";
        "Maltesische Lira",
        "displayName-count-other";
        "Maltesische Lira",
        "symbol";
        "MTL";
    }
    "MTP";
    {
        "displayName";
        "Maltesisches Pfund",
        "displayName-count-one";
        "Maltesische Pfund",
        "displayName-count-other";
        "Maltesische Pfund",
        "symbol";
        "MTP";
    }
    "MUR";
    {
        "displayName";
        "Mauritius-Rupie",
        "displayName-count-one";
        "Mauritius-Rupie",
        "displayName-count-other";
        "Mauritius-Rupien",
        "symbol";
    }

```

```

        "MUR",
        "symbol-alt-narrow";
        "Rs";
    }
    "MVP";
    {
        "displayName";
        "Malediven-Rupie (alt)",
        "displayName-count-one";
        "Malediven-Rupie (alt)",
        "displayName-count-other";
        "Malediven-Rupien (alt)";
    }
    "MVR";
    {
        "displayName";
        "Malediven-Rufiyaa",
        "displayName-count-one";
        "Malediven-Rufiyaa",
        "displayName-count-other";
        "Malediven-Rupien",
        "symbol";
        "MVR";
    }
    "MWK";
    {
        "displayName";
        "Malawi-Kwacha",
        "displayName-count-one";
        "Malawi-Kwacha",
        "displayName-count-other";
        "Malawi-Kwacha",
        "symbol";
        "MWK";
    }
    "MXN";
    {
        "displayName";
        "Mexikanischer Peso",
        "displayName-count-one";
        "Mexikanischer Peso",
        "displayName-count-other";
        "Mexikanische Pesos",
        "symbol";
        "MX$",
        "symbol-alt-narrow";
        "$";
    }
    "MXP";
    {
        "displayName";
        "Mexikanischer Silber-Peso (1861-1992)",
        "displayName-count-one";
        "Mexikanische Silber-Peso (1861-1992)",
        "displayName-count-other";
        "Mexikanische Silber-Pesos (1861-1992)",
        "symbol";
    }

```



```

        "MXP";
    }
    "MXV";
    {
        "displayName";
        "Mexicanischer Unidad de Inversion (UDI)",
        "displayName-count-one";
        "Mexicanischer Unidad de Inversion (UDI)",
        "displayName-count-other";
        "Mexikanische Unidad de Inversion (UDI)",
        "symbol";
        "MXV";
    }
    "MYR";
    {
        "displayName";
        "Malaysischer Ringgit",
        "displayName-count-one";
        "Malaysischer Ringgit",
        "displayName-count-other";
        "Malaysische Ringgit",
        "symbol";
        "MYR",
        "symbol-alt-narrow";
        "RM";
    }
    "MZE";
    {
        "displayName";
        "Mosambikanischer Escudo",
        "displayName-count-one";
        "Mozambikanische Escudo",
        "displayName-count-other";
        "Mozambikanische Escudo",
        "symbol";
        "MZE";
    }
    "MZM";
    {
        "displayName";
        "Mosambikanischer Metical (1980-2006)",
        "displayName-count-one";
        "Mosambikanischer Metical (1980-2006)",
        "displayName-count-other";
        "Mosambikanische Meticais (1980-2006)",
        "symbol";
        "MZM";
    }
    "MZN";
    {
        "displayName";
        "Mosambikanischer Metical",
        "displayName-count-one";
        "Mosambikanischer Metical",
        "displayName-count-other";
        "Mosambikanische Meticais",
        "symbol";
    }

```

```

        "MZN";
    }
    "NAD";
    {
        "displayName";
        "Namibia-Dollar",
        "displayName-count-one";
        "Namibia-Dollar",
        "displayName-count-other";
        "Namibia-Dollar",
        "symbol";
        "NAD",
        "symbol-alt-narrow";
        "$";
    }
    "NGN";
    {
        "displayName";
        "Nigerianischer Naira",
        "displayName-count-one";
        "Nigerianischer Naira",
        "displayName-count-other";
        "Nigerianische Naira",
        "symbol";
        "NGN",
        "symbol-alt-narrow";
        "₦";
    }
    "NIC";
    {
        "displayName";
        "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-one";
        "Nicaraguanischer Córdoba (1988-1991)",
        "displayName-count-other";
        "Nicaraguanische Córdoba (1988-1991)",
        "symbol";
        "NIC";
    }
    "NIO";
    {
        "displayName";
        "Nicaragua-Córdoba",
        "displayName-count-one";
        "Nicaragua-Córdoba",
        "displayName-count-other";
        "Nicaragua-Córdobas",
        "symbol";
        "NIO",
        "symbol-alt-narrow";
        "C$";
    }
    "NLG";
    {
        "displayName";
        "Niederländischer Gulden",
        "displayName-count-one";

```

```

        "Niederländischer Gulden",
        "displayName-count-other";
        "Niederländische Gulden",
        "symbol";
        "NLG";
    }
    "NOK";
    {
        "displayName";
        "Norwegische Krone",
        "displayName-count-one";
        "Norwegische Krone",
        "displayName-count-other";
        "Norwegische Kronen",
        "symbol";
        "NOK",
        "symbol-alt-narrow";
        "kr";
    }
    "NPR";
    {
        "displayName";
        "Nepalesische Rupie",
        "displayName-count-one";
        "Nepalesische Rupie",
        "displayName-count-other";
        "Nepalesische Rupien",
        "symbol";
        "NPR",
        "symbol-alt-narrow";
        "Rs";
    }
    "NZD";
    {
        "displayName";
        "Neuseeland-Dollar",
        "displayName-count-one";
        "Neuseeland-Dollar",
        "displayName-count-other";
        "Neuseeland-Dollar",
        "symbol";
        "NZ$",
        "symbol-alt-narrow";
        "$";
    }
    "OMR";
    {
        "displayName";
        "Omanischer Rial",
        "displayName-count-one";
        "Omanischer Rial",
        "displayName-count-other";
        "Omanische Rials",
        "symbol";
        "OMR";
    }
    "PAB";

```

```

    {
      "displayName";
      "Panamaischer Balboa",
      "displayName-count-one";
      "Panamaischer Balboa",
      "displayName-count-other";
      "Panamaische Balboas",
      "symbol";
      "PAB";
    }
    "PEI";
    {
      "displayName";
      "Peruanischer Inti",
      "displayName-count-one";
      "Peruanische Inti",
      "displayName-count-other";
      "Peruanische Inti",
      "symbol";
      "PEI";
    }
    "PEN";
    {
      "displayName";
      "Peruanischer Sol",
      "displayName-count-one";
      "Peruanischer Sol",
      "displayName-count-other";
      "Peruanische Sol",
      "symbol";
      "PEN";
    }
    "PES";
    {
      "displayName";
      "Peruanischer Sol (1863-1965)",
      "displayName-count-one";
      "Peruanischer Sol (1863-1965)",
      "displayName-count-other";
      "Peruanische Sol (1863-1965)",
      "symbol";
      "PES";
    }
    "PGK";
    {
      "displayName";
      "Papua-Neuguineischer Kina",
      "displayName-count-one";
      "Papua-Neuguineischer Kina",
      "displayName-count-other";
      "Papua-Neuguineische Kina",
      "symbol";
      "PGK";
    }
    "PHP";
    {
      "displayName";

```

```

        "Philippinischer Peso",
        "displayName-count-one";
        "Philippinischer Peso",
        "displayName-count-other";
        "Philippinische Pesos",
        "symbol";
        "PHP",
        "symbol-alt-narrow";
        "₱";
    }
    "PKR";
    {
        "displayName";
        "Pakistanische Rupie",
        "displayName-count-one";
        "Pakistanische Rupie",
        "displayName-count-other";
        "Pakistanische Rupien",
        "symbol";
        "PKR",
        "symbol-alt-narrow";
        "Rs";
    }
    "PLN";
    {
        "displayName";
        "Polnischer Złoty",
        "displayName-count-one";
        "Polnischer Złoty",
        "displayName-count-other";
        "Polnische Złoty",
        "symbol";
        "PLN",
        "symbol-alt-narrow";
        "zł";
    }
    "PLZ";
    {
        "displayName";
        "Polnischer Zloty (1950-1995)",
        "displayName-count-one";
        "Polnischer Zloty (1950-1995)",
        "displayName-count-other";
        "Polnische Zloty (1950-1995)",
        "symbol";
        "PLZ";
    }
    "PTE";
    {
        "displayName";
        "Portugiesischer Escudo",
        "displayName-count-one";
        "Portugiesische Escudo",
        "displayName-count-other";
        "Portugiesische Escudo",
        "symbol";
        "PTE";
    }

```

```

    }
    "PYG";
    {
        "displayName";
        "Paraguayischer Guaraní",
        "displayName-count-one";
        "Paraguayischer Guaraní",
        "displayName-count-other";
        "Paraguayische Guaraníes",
        "symbol";
        "PYG",
        "symbol-alt-narrow";
        "₲";
    }
    "QAR";
    {
        "displayName";
        "Katar-Riyal",
        "displayName-count-one";
        "Katar-Riyal",
        "displayName-count-other";
        "Katar-Riyal",
        "symbol";
        "QAR";
    }
    "RHD";
    {
        "displayName";
        "Rhodesischer Dollar",
        "displayName-count-one";
        "Rhodesische Dollar",
        "displayName-count-other";
        "Rhodesische Dollar",
        "symbol";
        "RHD";
    }
    "ROL";
    {
        "displayName";
        "Rumänischer Leu (1952-2006)",
        "displayName-count-one";
        "Rumänischer Leu (1952-2006)",
        "displayName-count-other";
        "Rumänische Leu (1952-2006)",
        "symbol";
        "ROL";
    }
    "RON";
    {
        "displayName";
        "Rumänischer Leu",
        "displayName-count-one";
        "Rumänischer Leu",
        "displayName-count-other";
        "Rumänische Leu",
        "symbol";
        "RON";
    }

```

```

        "symbol-alt-narrow";
        "L";
    }
    "RSD";
    {
        "displayName";
        "Serbischer Dinar",
        "displayName-count-one";
        "Serbischer Dinar",
        "displayName-count-other";
        "Serbische Dinaren",
        "symbol";
        "RSD";
    }
    "RUB";
    {
        "displayName";
        "Russischer Rubel",
        "displayName-count-one";
        "Russischer Rubel",
        "displayName-count-other";
        "Russische Rubel",
        "symbol";
        "RUB",
        "symbol-alt-narrow";
        "₽";
    }
    "RUR";
    {
        "displayName";
        "Russischer Rubel (1991-1998)",
        "displayName-count-one";
        "Russischer Rubel (1991-1998)",
        "displayName-count-other";
        "Russische Rubel (1991-1998)",
        "symbol";
        "RUR",
        "symbol-alt-narrow";
        "p.";
    }
    "RWF";
    {
        "displayName";
        "Ruanda-Franc",
        "displayName-count-one";
        "Ruanda-Franc",
        "displayName-count-other";
        "Ruanda-Francis",
        "symbol";
        "RWF",
        "symbol-alt-narrow";
        "F.Rw";
    }
    "SAR";
    {
        "displayName";
        "Saudi-Rial",

```

```

        "displayName-count-one";
        "Saudi-Rial",
        "displayName-count-other";
        "Saudi-Rial",
        "symbol";
        "SAR";
    }
    "SBD";
    {
        "displayName";
        "Salomonen-Dollar",
        "displayName-count-one";
        "Salomonen-Dollar",
        "displayName-count-other";
        "Salomonen-Dollar",
        "symbol";
        "SBD",
        "symbol-alt-narrow";
        "$";
    }
    "SCR";
    {
        "displayName";
        "Seychellen-Rupie",
        "displayName-count-one";
        "Seychellen-Rupie",
        "displayName-count-other";
        "Seychellen-Rupien",
        "symbol";
        "SCR";
    }
    "SDD";
    {
        "displayName";
        "Sudanesischer Dinar (1992-2007)",
        "displayName-count-one";
        "Sudanesischer Dinar (1992-2007)",
        "displayName-count-other";
        "Sudanesische Dinar (1992-2007)",
        "symbol";
        "SDD";
    }
    "SDG";
    {
        "displayName";
        "Sudanesisches Pfund",
        "displayName-count-one";
        "Sudanesisches Pfund",
        "displayName-count-other";
        "Sudanesische Pfund",
        "symbol";
        "SDG";
    }
    "SDP";
    {
        "displayName";
        "Sudanesisches Pfund (1957-1998)",

```



```

        "displayName-count-one";
        "Sudanesisches Pfund (1957-1998)",
        "displayName-count-other";
        "Sudanesische Pfund (1957-1998)",
        "symbol";
        "SDP";
    }
    "SEK";
    {
        "displayName";
        "Schwedische Krone",
        "displayName-count-one";
        "Schwedische Krone",
        "displayName-count-other";
        "Schwedische Kronen",
        "symbol";
        "SEK",
        "symbol-alt-narrow";
        "kr";
    }
    "SGD";
    {
        "displayName";
        "Singapur-Dollar",
        "displayName-count-one";
        "Singapur-Dollar",
        "displayName-count-other";
        "Singapur-Dollar",
        "symbol";
        "SGD",
        "symbol-alt-narrow";
        "$";
    }
    "SHP";
    {
        "displayName";
        "St. Helena-Pfund",
        "displayName-count-one";
        "St. Helena-Pfund",
        "displayName-count-other";
        "St. Helena-Pfund",
        "symbol";
        "SHP",
        "symbol-alt-narrow";
        "£";
    }
    "SIT";
    {
        "displayName";
        "Slowenischer Tolar",
        "displayName-count-one";
        "Slowenischer Tolar",
        "displayName-count-other";
        "Slowenische Tolar",
        "symbol";
        "SIT";
    }
}

```

```

"SKK";
{
  "displayName";
  "Slowakische Krone",
    "displayName-count-one";
  "Slowakische Kronen",
    "displayName-count-other";
  "Slowakische Kronen",
    "symbol";
  "SKK";
}
"SLL";
{
  "displayName";
  "Sierra-leonischer Leone",
    "displayName-count-one";
  "Sierra-leonischer Leone",
    "displayName-count-other";
  "Sierra-leonische Leones",
    "symbol";
  "SLL";
}
"SOS";
{
  "displayName";
  "Somalia-Schilling",
    "displayName-count-one";
  "Somalia-Schilling",
    "displayName-count-other";
  "Somalia-Schilling",
    "symbol";
  "SOS";
}
"SRD";
{
  "displayName";
  "Suriname-Dollar",
    "displayName-count-one";
  "Suriname-Dollar",
    "displayName-count-other";
  "Suriname-Dollar",
    "symbol";
  "SRD",
    "symbol-alt-narrow";
  "$";
}
"SRG";
{
  "displayName";
  "Suriname Gulden",
    "displayName-count-one";
  "Suriname-Gulden",
    "displayName-count-other";
  "Suriname-Gulden",
    "symbol";
  "SRG";
}

```

```

"SSP";
{
  "displayName";
  "Südsudanesisches Pfund",
    "displayName-count-one";
  "Südsudanesisches Pfund",
    "displayName-count-other";
  "Südsudanesische Pfund",
    "symbol";
  "SSP",
    "symbol-alt-narrow";
  "£";
}
"STD";
{
  "displayName";
  "São-toméischer Dobra",
    "displayName-count-one";
  "São-toméischer Dobra",
    "displayName-count-other";
  "São-toméische Dobra",
    "symbol";
  "STD",
    "symbol-alt-narrow";
  "Db";
}
"SUR";
{
  "displayName";
  "Sowjetischer Rubel",
    "displayName-count-one";
  "Sowjetische Rubel",
    "displayName-count-other";
  "Sowjetische Rubel",
    "symbol";
  "SUR";
}
"SVC";
{
  "displayName";
  "El Salvador Colon",
    "displayName-count-one";
  "El Salvador-Colon",
    "displayName-count-other";
  "El Salvador-Colon",
    "symbol";
  "SVC";
}
"SYP";
{
  "displayName";
  "Syrisches Pfund",
    "displayName-count-one";
  "Syrisches Pfund",
    "displayName-count-other";
  "Syrische Pfund",
    "symbol";
}

```

```

        "SYP",
        "symbol-alt-narrow";
        "SYP";
    }
    "SZL";
    {
        "displayName";
        "Swasiländischer Lilangeni",
        "displayName-count-one";
        "Swasiländischer Lilangeni",
        "displayName-count-other";
        "Swasiländische Emalangeni",
        "symbol";
        "SZL";
    }
    "THB";
    {
        "displayName";
        "Thailändischer Baht",
        "displayName-count-one";
        "Thailändischer Baht",
        "displayName-count-other";
        "Thailändische Baht",
        "symbol";
        "฿",
        "symbol-alt-narrow";
        "฿";
    }
    "TJR";
    {
        "displayName";
        "Tadschikistan Rubel",
        "displayName-count-one";
        "Tadschikistan-Rubel",
        "displayName-count-other";
        "Tadschikistan-Rubel",
        "symbol";
        "TJR";
    }
    "TJS";
    {
        "displayName";
        "Tadschikistan-Somoni",
        "displayName-count-one";
        "Tadschikistan-Somoni",
        "displayName-count-other";
        "Tadschikistan-Somoni",
        "symbol";
        "TJS";
    }
    "TMM";
    {
        "displayName";
        "Turkmenistan-Manat (1993-2009)",
        "displayName-count-one";
        "Turkmenistan-Manat (1993-2009)",
        "displayName-count-other";
    }

```

```

        "Turkmenistan-Manat (1993-2009)",
        "symbol";
        "TMM";
    }
    "TMT";
    {
        "displayName";
        "Turkmenistan-Manat",
        "displayName-count-one";
        "Turkmenistan-Manat",
        "displayName-count-other";
        "Turkmenistan-Manat",
        "symbol";
        "TMT";
    }
    "TND";
    {
        "displayName";
        "Tunesischer Dinar",
        "displayName-count-one";
        "Tunesischer Dinar",
        "displayName-count-other";
        "Tunesische Dinar",
        "symbol";
        "TND";
    }
    "TOP";
    {
        "displayName";
        "Tongaischer Pa'anga",
        "displayName-count-one";
        "Tongaischer Pa'anga",
        "displayName-count-other";
        "Tongaische Pa'anga",
        "symbol";
        "TOP",
        "symbol-alt-narrow";
        "T$";
    }
    "TPE";
    {
        "displayName";
        "Timor-Escudo",
        "displayName-count-one";
        "Timor-Escudo",
        "displayName-count-other";
        "Timor-Escudo",
        "symbol";
        "TPE";
    }
    "TRL";
    {
        "displayName";
        "Türkische Lira (1922-2005)",
        "displayName-count-one";
        "Türkische Lira (1922-2005)",
        "displayName-count-other";
    }

```

```

        "Türkische Lira (1922-2005)",
        "symbol";
        "TRL";
    }
    "TRY";
    {
        "displayName";
        "Türkische Lira",
        "displayName-count-one";
        "Türkische Lira",
        "displayName-count-other";
        "Türkische Lira",
        "symbol";
        "TRY",
        "symbol-alt-narrow";
        "₺",
        "symbol-alt-variant";
        "TL";
    }
    "TTD";
    {
        "displayName";
        "Trinidad und Tobago-Dollar",
        "displayName-count-one";
        "Trinidad und Tobago-Dollar",
        "displayName-count-other";
        "Trinidad und Tobago-Dollar",
        "symbol";
        "TTD",
        "symbol-alt-narrow";
        "$";
    }
    "TWD";
    {
        "displayName";
        "Neuer Taiwan-Dollar",
        "displayName-count-one";
        "Neuer Taiwan-Dollar",
        "displayName-count-other";
        "Neue Taiwan-Dollar",
        "symbol";
        "NT$",
        "symbol-alt-narrow";
        "NT$";
    }
    "TZS";
    {
        "displayName";
        "Tansania-Schilling",
        "displayName-count-one";
        "Tansania-Schilling",
        "displayName-count-other";
        "Tansania-Schilling",
        "symbol";
        "TZS";
    }
    "UAH";

```

```

    {
      "displayName";
      "Ukrainische Hrywnja",
      "displayName-count-one";
      "Ukrainische Hrywnja",
      "displayName-count-other";
      "Ukrainische Hrywen",
      "symbol";
      "UAH",
      "symbol-alt-narrow";
      "₴";
    }
    "UAK";
    {
      "displayName";
      "Ukrainischer Karbovanetz",
      "displayName-count-one";
      "Ukrainische Karbovanetz",
      "displayName-count-other";
      "Ukrainische Karbovanetz",
      "symbol";
      "UAK";
    }
    "UGS";
    {
      "displayName";
      "Uganda-Schilling (1966-1987)",
      "displayName-count-one";
      "Uganda-Schilling (1966-1987)",
      "displayName-count-other";
      "Uganda-Schilling (1966-1987)",
      "symbol";
      "UGS";
    }
    "UGX";
    {
      "displayName";
      "Uganda-Schilling",
      "displayName-count-one";
      "Uganda-Schilling",
      "displayName-count-other";
      "Uganda-Schilling",
      "symbol";
      "UGX";
    }
    "USD";
    {
      "displayName";
      "US-Dollar",
      "displayName-count-one";
      "US-Dollar",
      "displayName-count-other";
      "US-Dollar",
      "symbol";
      "$",
      "symbol-alt-narrow";
      "$";
    }

```

```

    }
    "USN";
    {
        "displayName";
        "US Dollar (Nächster Tag)",
        "displayName-count-one";
        "US-Dollar (Nächster Tag)",
        "displayName-count-other";
        "US-Dollar (Nächster Tag)",
        "symbol";
        "USN";
    }
    "USS";
    {
        "displayName";
        "US Dollar (Gleicher Tag)",
        "displayName-count-one";
        "US-Dollar (Gleicher Tag)",
        "displayName-count-other";
        "US-Dollar (Gleicher Tag)",
        "symbol";
        "USS";
    }
    "UYI";
    {
        "displayName";
        "Uruguayischer Peso (Indexierte Rechnungseinheiten)",
        "displayName-count-one";
        "Uruguayischer Peso (Indexierte Rechnungseinheiten)",
        "displayName-count-other";
        "Uruguayische Pesos (Indexierte Rechnungseinheiten)",
        "symbol";
        "UYI";
    }
    "UYP";
    {
        "displayName";
        "Uruguayischer Peso (1975-1993)",
        "displayName-count-one";
        "Uruguayischer Peso (1975-1993)",
        "displayName-count-other";
        "Uruguayische Pesos (1975-1993)",
        "symbol";
        "UYP";
    }
    "UYU";
    {
        "displayName";
        "Uruguayischer Peso",
        "displayName-count-one";
        "Uruguayischer Peso",
        "displayName-count-other";
        "Uruguayische Pesos",
        "symbol";
        "UYU",
        "symbol-alt-narrow";
        "$";
    }

```



```

    }
    "UZS";
    {
        "displayName";
        "Usbekistan-Sum",
        "displayName-count-one";
        "Usbekistan-Sum",
        "displayName-count-other";
        "Usbekistan-Sum",
        "symbol";
        "UZS";
    }
    "VEB";
    {
        "displayName";
        "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-one";
        "Venezolanischer Bolívar (1871-2008)",
        "displayName-count-other";
        "Venezolanische Bolívares (1871-2008)",
        "symbol";
        "VEB";
    }
    "VEF";
    {
        "displayName";
        "Venezolanischer Bolívar",
        "displayName-count-one";
        "Venezolanischer Bolívar",
        "displayName-count-other";
        "Venezolanische Bolívares",
        "symbol";
        "VEF",
        "symbol-alt-narrow";
        "Bs";
    }
    "VND";
    {
        "displayName";
        "Vietnamesischer Dong",
        "displayName-count-one";
        "Vietnamesischer Dong",
        "displayName-count-other";
        "Vietnamesische Dong",
        "symbol";
        "₫",
        "symbol-alt-narrow";
        "₫";
    }
    "VNN";
    {
        "displayName";
        "Vietnamesischer Dong (1978-1985)",
        "displayName-count-one";
        "Vietnamesischer Dong (1978-1985)",
        "displayName-count-other";
        "Vietnamesische Dong (1978-1985)",

```

```

        "symbol";
        "VNN";
    }
    "VUV";
    {
        "displayName";
        "Vanuatu-Vatu",
        "displayName-count-one";
        "Vanuatu-Vatu",
        "displayName-count-other";
        "Vanuatu-Vatu",
        "symbol";
        "VUV";
    }
    "WST";
    {
        "displayName";
        "Samoanischer Tala",
        "displayName-count-one";
        "Samoanischer Tala",
        "displayName-count-other";
        "Samoanische Tala",
        "symbol";
        "WST";
    }
    "XAF";
    {
        "displayName";
        "CFA-Franc (BEAC)",
        "displayName-count-one";
        "CFA-Franc (BEAC)",
        "displayName-count-other";
        "CFA-Franc (BEAC)",
        "symbol";
        "FCFA";
    }
    "XAG";
    {
        "displayName";
        "Unze Silber",
        "displayName-count-one";
        "Unze Silber",
        "displayName-count-other";
        "Unzen Silber",
        "symbol";
        "XAG";
    }
    "XAU";
    {
        "displayName";
        "Unze Gold",
        "displayName-count-one";
        "Unze Gold",
        "displayName-count-other";
        "Unzen Gold",
        "symbol";
        "XAU";
    }

```

```

    }
    "XBA";
    {
        "displayName";
        "Europäische Rechnungseinheit",
        "displayName-count-one";
        "Europäische Rechnungseinheiten",
        "displayName-count-other";
        "Europäische Rechnungseinheiten",
        "symbol";
        "XBA";
    }
    "XBB";
    {
        "displayName";
        "Europäische Währungseinheit (XBB)",
        "displayName-count-one";
        "Europäische Währungseinheiten (XBB)",
        "displayName-count-other";
        "Europäische Währungseinheiten (XBB)",
        "symbol";
        "XBB";
    }
    "XBC";
    {
        "displayName";
        "Europäische Rechnungseinheit (XBC)",
        "displayName-count-one";
        "Europäische Rechnungseinheiten (XBC)",
        "displayName-count-other";
        "Europäische Rechnungseinheiten (XBC)",
        "symbol";
        "XBC";
    }
    "XBD";
    {
        "displayName";
        "Europäische Rechnungseinheit (XBD)",
        "displayName-count-one";
        "Europäische Rechnungseinheiten (XBD)",
        "displayName-count-other";
        "Europäische Rechnungseinheiten (XBD)",
        "symbol";
        "XBD";
    }
    "XCD";
    {
        "displayName";
        "Ostkaribischer Dollar",
        "displayName-count-one";
        "Ostkaribischer Dollar",
        "displayName-count-other";
        "Ostkaribische Dollar",
        "symbol";
        "EC$",
        "symbol-alt-narrow";
        "$";
    }

```

```

    }
    "XDR";
    {
        "displayName";
        "Sonderziehungsrechte",
            "displayName-count-one";
        "Sonderziehungsrechte",
            "displayName-count-other";
        "Sonderziehungsrechte",
            "symbol";
        "XDR";
    }
    "XEU";
    {
        "displayName";
        "Europäische Währungseinheit (XEU)",
            "displayName-count-one";
        "Europäische Währungseinheiten (XEU)",
            "displayName-count-other";
        "Europäische Währungseinheiten (XEU)",
            "symbol";
        "XEU";
    }
    "XFO";
    {
        "displayName";
        "Französischer Gold-Franc",
            "displayName-count-one";
        "Französische Gold-Franc",
            "displayName-count-other";
        "Französische Gold-Franc",
            "symbol";
        "XFO";
    }
    "XFU";
    {
        "displayName";
        "Französischer UIC-Franc",
            "displayName-count-one";
        "Französische UIC-Franc",
            "displayName-count-other";
        "Französische UIC-Franc",
            "symbol";
        "XFU";
    }
    "XOF";
    {
        "displayName";
        "CFA-Franc (BCEAO)",
            "displayName-count-one";
        "CFA-Franc (BCEAO)",
            "displayName-count-other";
        "CFA-Francs (BCEAO)",
            "symbol";
        "CFA";
    }
    "XPD";

```

```

    {
      "displayName";
      "Unze Palladium",
      "displayName-count-one";
      "Unze Palladium",
      "displayName-count-other";
      "Unzen Palladium",
      "symbol";
      "XPD";
    }
    "XPF";
    {
      "displayName";
      "CFP-Franc",
      "displayName-count-one";
      "CFP-Franc",
      "displayName-count-other";
      "CFP-Franc",
      "symbol";
      "CFPF";
    }
    "XPT";
    {
      "displayName";
      "Unze Platin",
      "displayName-count-one";
      "Unze Platin",
      "displayName-count-other";
      "Unzen Platin",
      "symbol";
      "XPT";
    }
    "XRE";
    {
      "displayName";
      "RINET Funds",
      "displayName-count-one";
      "RINET Funds",
      "displayName-count-other";
      "RINET Funds",
      "symbol";
      "XRE";
    }
    "XSU";
    {
      "displayName";
      "SUCRE",
      "displayName-count-one";
      "SUCRE",
      "displayName-count-other";
      "SUCRE",
      "symbol";
      "XSU";
    }
    "XTS";
    {
      "displayName";

```

```

        "Testwährung",
        "displayName-count-one";
        "Testwährung",
        "displayName-count-other";
        "Testwährung",
        "symbol";
        "XTS";
    }
    "XUA";
    {
        "displayName";
        "Rechnungseinheit der AfEB",
        "displayName-count-one";
        "Rechnungseinheit der AfEB",
        "displayName-count-other";
        "Rechnungseinheiten der AfEB",
        "symbol";
        "XUA";
    }
    "XXX";
    {
        "displayName";
        "Unbekannte Währung",
        "displayName-count-one";
        "(unbekannte Währung)",
        "displayName-count-other";
        "(unbekannte Währung)",
        "symbol";
        "XXX";
    }
    "YDD";
    {
        "displayName";
        "Jemen-Dinar",
        "displayName-count-one";
        "Jemen-Dinar",
        "displayName-count-other";
        "Jemen-Dinar",
        "symbol";
        "YDD";
    }
    "YER";
    {
        "displayName";
        "Jemen-Rial",
        "displayName-count-one";
        "Jemen-Rial",
        "displayName-count-other";
        "Jemen-Rial",
        "symbol";
        "YER";
    }
    "YUD";
    {
        "displayName";
        "Jugoslawischer Dinar (1966-1990)",
        "displayName-count-one";
    }

```

```

        "Jugoslawischer Dinar (1966-1990)",
        "displayName-count-other";
        "Jugoslawische Dinar (1966-1990)",
        "symbol";
        "YUD";
    }
    "YUM";
    {
        "displayName";
        "Jugoslawischer Neuer Dinar (1994-2002)",
        "displayName-count-one";
        "Jugoslawischer Neuer Dinar (1994-2002)",
        "displayName-count-other";
        "Jugoslawische Neue Dinar (1994-2002)",
        "symbol";
        "YUM";
    }
    "YUN";
    {
        "displayName";
        "Jugoslawischer Dinar (konvertibel)",
        "displayName-count-one";
        "Jugoslawische Dinar (konvertibel)",
        "displayName-count-other";
        "Jugoslawische Dinar (konvertibel)",
        "symbol";
        "YUN";
    }
    "YUR";
    {
        "displayName";
        "Jugoslawischer reformierter Dinar (1992-1993)",
        "displayName-count-one";
        "Jugoslawischer reformierter Dinar (1992-1993)",
        "displayName-count-other";
        "Jugoslawische reformierte Dinar (1992-1993)",
        "symbol";
        "YUR";
    }
    "ZAL";
    {
        "displayName";
        "Südafrikanischer Rand (Finanz)",
        "displayName-count-one";
        "Südafrikanischer Rand (Finanz)",
        "displayName-count-other";
        "Südafrikanischer Rand (Finanz)",
        "symbol";
        "ZAL";
    }
    "ZAR";
    {
        "displayName";
        "Südafrikanischer Rand",
        "displayName-count-one";
        "Südafrikanischer Rand",
        "displayName-count-other";
    }

```

```

        "Südafrikanische Rand",
        "symbol";
    "ZAR",
        "symbol-alt-narrow";
    "R";
}
"ZMK";
{
    "displayName";
    "Kwacha (1968-2012)",
        "displayName-count-one";
    "Kwacha (1968-2012)",
        "displayName-count-other";
    "Kwacha (1968-2012)",
        "symbol";
    "ZMK";
}
"ZMW";
{
    "displayName";
    "Kwacha",
        "displayName-count-one";
    "Kwacha",
        "displayName-count-other";
    "Kwacha",
        "symbol";
    "ZMW",
        "symbol-alt-narrow";
    "K";
}
"ZRN";
{
    "displayName";
    "Zaire-Neuer Zaïre (1993-1998)",
        "displayName-count-one";
    "Zaire-Neuer Zaïre (1993-1998)",
        "displayName-count-other";
    "Zaire-Neue Zaïre (1993-1998)",
        "symbol";
    "ZRN";
}
"ZRZ";
{
    "displayName";
    "Zaire-Zaïre (1971-1993)",
        "displayName-count-one";
    "Zaire-Zaïre (1971-1993)",
        "displayName-count-other";
    "Zaire-Zaïre (1971-1993)",
        "symbol";
    "ZRZ";
}
"ZWD";
{
    "displayName";
    "Simbabwe-Dollar (1980-2008)",
        "displayName-count-one";

```



```

        "Simbabwe-Dollar (1980-2008)",
            "displayName-count-other";
    "Simbabwe-Dollar (1980-2008)",
        "symbol";
    "ZWD";
}
"ZWL";
{
    "displayName";
    "Simbabwe-Dollar (2009)",
        "displayName-count-one";
    "Simbabwe-Dollar (2009)",
        "displayName-count-other";
    "Simbabwe-Dollar (2009)",
        "symbol";
    "ZWL";
}
"ZWR";
{
    "displayName";
    "Simbabwe-Dollar (2008)",
        "displayName-count-one";
    "Simbabwe-Dollar (2008)",
        "displayName-count-other";
    "Simbabwe-Dollar (2008)",
        "symbol";
    "ZWR";
}
}
}
}
}

```

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
//Load the L10n, loadCldr from ej2-base
import { loadCldr, L10n } from '@syncfusion/ej2-base';
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
//load the CLDR data files.
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as deTimeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, deTimeZoneNames);
L10n.load({
    'de': {
        'daterangepicker': {
            applyText: 'Sich bewerben',
            cancelText: 'Stornieren',
            customRange: 'benutzerdefinierten Bereich',
            days: 'Tage',
            endLabel: 'Wählen Sie Enddatum',
        }
    }
});
```

```

        placeholder: 'Wählen Sie einen Bereich aus',
        selectedDays: 'Ausgewählte Tage',
        startLabel: 'Wählen Sie Startdatum'
    }
}
});
//import the daterangepicker component
function App() {
    return <DateRangePickerComponent id="daterangepicker" locale='de' />;
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
//Load the L10n, loadCldr from ej2-base
import { loadCldr, L10n } from '@syncfusion/ej2-base';
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
//load the CLDR data files.
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as deTimeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, deTimeZoneNames);
L10n.load({
    'de': {
        'daterangepicker': {
            applyText: 'Sich bewerben',
            cancelText: 'Stornieren',
            customRange: 'benutzerdefinierten Bereich',
            days: 'Tage',
            endLabel: 'Wählen Sie Enddatum',
            placeholder: 'Wählen Sie einen Bereich aus',
            selectedDays: 'Ausgewählte Tage',
            startLabel: 'Wählen Sie Startdatum'
        }
    }
});
//import the daterangepicker component
function App() {
    return <DateRangePickerComponent id="daterangepicker" locale='de' />;
};
ReactDOM.render(<App />, document.getElementById('element'));

```

NUMBERINGSYSTEMS.JSON

```

{
  "supplemental": {
    "version": {
      "_number": "$Revision: 12732 $",
      "_unicodeVersion": "9.0.0",
      "_cldrVersion": "31"
    },

```

```

"numberingSystems": {
  "adlm": {
    "_digits": "ᲐᲑᲒᲓᲔᲕᲖᲗᲘᲙ",
    "_type": "numeric"
  },
  "ahom": {
    "_digits": "ᩌᩍᩎᩏᩐᩑᩒᩓᩔᩕᩖ",
    "_type": "numeric"
  },
  "arab": {
    "_digits": "٠١٢٣٤٥٦٧٨٩",
    "_type": "numeric"
  },
  "arabext": {
    "_digits": "٠١٢٣٤٥٦٧٨٩",
    "_type": "numeric"
  },
  "armn": {
    "_rules": "armenian-upper",
    "_type": "algorithmic"
  },
  "armnlow": {
    "_rules": "armenian-lower",
    "_type": "algorithmic"
  },
  "bali": {
    "_digits": "ᬀᬁᬂᬃᬄᬅᬆᬇᬈᬉ",
    "_type": "numeric"
  },
  "beng": {
    "_digits": "০১২৩৪৫৬৭৮৯",
    "_type": "numeric"
  },
  "bhks": {
    "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊",
    "_type": "numeric"
  },
  "brah": {
    "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉",
    "_type": "numeric"
  },
  "cakm": {
    "_digits": "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
    "_type": "numeric"
  },
  "cham": {
    "_digits": "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
    "_type": "numeric"
  },
  "cyr1": {
    "_rules": "cyrillic-lower",
    "_type": "algorithmic"
  },
  "deva": {
    "_digits": "०१२३४५६७८९",
    "_type": "numeric"
  }
}

```

```
,
"ethi": {
  "_rules": "ethiopic",
  "_type": "algorithmic"
},
"fullwide": {
  "_digits": "0 1 2 3 4 5 6 7 8 9",
  "_type": "numeric"
},
"geor": {
  "_rules": "georgian",
  "_type": "algorithmic"
},
"grek": {
  "_rules": "greek-upper",
  "_type": "algorithmic"
},
"greklow": {
  "_rules": "greek-lower",
  "_type": "algorithmic"
},
"gujr": {
  "_digits": "૦ ૧ ૨ ૩ ૪ ૫ ૬ ૭ ૮ ૯",
  "_type": "numeric"
},
"guru": {
  "_digits": "੦ ੧ ੨ ੩ ੪ ੫ ੬ ੭ ੮ ੯",
  "_type": "numeric"
},
"hanidays": {
  "_rules": "zh/SpelloutRules/spellout-numbering-days",
  "_type": "algorithmic"
},
"hanidec": {
  "_digits": "〇 一 二 三 四 五 六 七 八 九",
  "_type": "numeric"
},
"hans": {
  "_rules": "zh/SpelloutRules/spellout-cardinal",
  "_type": "algorithmic"
},
"hansfin": {
  "_rules": "zh/SpelloutRules/spellout-cardinal-financial",
  "_type": "algorithmic"
},
"hant": {
  "_rules": "zh_Hant/SpelloutRules/spellout-cardinal",
  "_type": "algorithmic"
},
"hantfin": {
  "_rules": "zh_Hant/SpelloutRules/spellout-cardinal-financial",
  "_type": "algorithmic"
},
"hebr": {
  "_rules": "hebrew",
  "_type": "algorithmic"
}
```

```

{
  "hmng": {
    "_digits": "0000000000",
    "_type": "numeric"
  },
  "java": {
    "_digits": "၀၀၀၀၀၀၀၀၀၀၀၀၀၀၀၀",
    "_type": "numeric"
  },
  "jpan": {
    "_rules": "ja/SpelloutRules/spellout-cardinal",
    "_type": "algorithmic"
  },
  "jpanfin": {
    "_rules": "ja/SpelloutRules/spellout-cardinal-financial",
    "_type": "algorithmic"
  },
  "kali": {
    "_digits": "0000000000",
    "_type": "numeric"
  },
  "khmr": {
    "_digits": "០១២៣៤៥៦៧៨៩",
    "_type": "numeric"
  },
  "knda": {
    "_digits": "೦೧೨೩೪೫೬೭೮೯",
    "_type": "numeric"
  },
  "lana": {
    "_digits": "0000000000",
    "_type": "numeric"
  },
  "lanatham": {
    "_digits": "0000000000",
    "_type": "numeric"
  },
  "lao": {
    "_digits": "໐໑໒໓໔໕໖໗໘໙",
    "_type": "numeric"
  },
  "latn": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "lepc": {
    "_digits": "0000000000",
    "_type": "numeric"
  },
  "limb": {
    "_digits": "0000000000",
    "_type": "numeric"
  },
  "mathbold": {
    "digits": "0123456789",

```

```

    "_type": "numeric"
  },
  "mathdbl": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mathmono": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mathsanb": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mathsans": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mlym": {
    "_digits": "൦൧൨൩൪൫൬൭൮൯",
    "_type": "numeric"
  },
  "modi": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "mong": {
    "_digits": "᠐᠑᠒᠓᠐ᠠᠨᠨᠠᠭᠤᠯ",
    "_type": "numeric"
  },
  "mroo": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "mtei": {
    "_digits": "ႤႬႭᆪᆫᆯᆰᆱᆲᆳᆴᆵᆶᆷᆸᆹᆺᆻᆼᆽᆾᆿ",
    "_type": "numeric"
  },
  "mymr": {
    "_digits": "၀၁၂၃၄၅၆၇၈၉",
    "_type": "numeric"
  },
  "mymrshan": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mymrtlng": {
    "_digits": "ႤႬᆪᆫᆯᆰᆱᆲᆳᆴᆵᆶᆷᆸᆹᆺᆻᆼᆽᆾᆿ",
    "_type": "numeric"
  },
  "newa": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  },

```

```
{
  "nko": {
    "_digits": "ᠨᠠᠭᠤᠯᠠᠩᠭ᠋ᠣᠨ",
    "_type": "numeric"
  },
  "olck": {
    "_digits": "ᠣᠯᠡᠴᠬᠡ",
    "_type": "numeric"
  },
  "orya": {
    "_digits": "ଓଡ଼ିଆ",
    "_type": "numeric"
  },
  "osma": {
    "_digits": "ᠣᠰᠢᠮ᠎ᠠ",
    "_type": "numeric"
  },
  "roman": {
    "_rules": "roman-upper",
    "_type": "algorithmic"
  },
  "romanlow": {
    "_rules": "roman-lower",
    "_type": "algorithmic"
  },
  "saur": {
    "_digits": "ᱵᱟᱹᱨᱫᱽᱯᱪᱟ",
    "_type": "numeric"
  },
  "shrd": {
    "_digits": "ᱥᱚᱱᱚᱛ",
    "_type": "numeric"
  },
  "sind": {
    "_digits": "ᱥᱤᱨᱫᱷᱟᱹᱭ",
    "_type": "numeric"
  },
  "sinh": {
    "_digits": "සිංහල",
    "_type": "numeric"
  },
  "sora": {
    "_digits": "ᱥᱟᱱᱜᱟᱴ",
    "_type": "numeric"
  },
  "sund": {
    "_digits": "ᱦᱚᱱᱚᱛ",
    "_type": "numeric"
  },
  "takr": {
    "_digits": "ᱞᱟᱹᱨᱫᱽᱯᱪᱟ",
    "_type": "numeric"
  },
  "tal": {
    "_digits": "ᱠᱚᱱᱚᱛ",
    "_type": "numeric"
  }
}
```

```

    },
    "taml": {
      "_rules": "tamil",
      "_type": "algorithmic"
    },
    "tamldec": {
      "_digits": "0௧௨௩௪௫௬௭௮௯",
      "_type": "numeric"
    },
    "telu": {
      "_digits": "౦౧౨౩౪౫౬౭౮౯",
      "_type": "numeric"
    },
    "thai": {
      "_digits": "๐๑๒๓๔๕๖๗๘๙",
      "_type": "numeric"
    },
    "tibtb": {
      "_digits": "༠༡༢༣༤༥༦༧༨༩",
      "_type": "numeric"
    },
    "tirh": {
      "_digits": "୦୧୨୩୪୫୬୭୮୯",
      "_type": "numeric"
    },
    "vaih": {
      "_digits": "0123456789",
      "_type": "numeric"
    },
    "wara": {
      "_digits": "0123456789",
      "_type": "numeric"
    }
  }
}

```

NUMBERINGSYSTEMS.JSX

```

{
  "supplemental";
  {
    "version";
    {
      "_number";
      "$Revision: 12732 $",
      "_unicodeVersion";
      "9.0.0",
      "_cldrVersion";
      "31";
    }
    "numberingSystems";
    {
      "adlm";
      {

```



```

        "_digits";
        "୧୨୩୪୫୬୭୮୯୦",
        "_type";
        "numeric";
    }
    "ahom";
    {
        "_digits";
        "ᱠᱡᱢᱣᱤᱨᱫᱽᱴᱚᱴ",
        "_type";
        "numeric";
    }
    "arab";
    {
        "_digits";
        "٠١٢٣٤٥٦٧٨٩",
        "_type";
        "numeric";
    }
    "arabext";
    {
        "_digits";
        "٠١٢٣٤٥٦٧٨٩",
        "_type";
        "numeric";
    }
    "armn";
    {
        "_rules";
        "armenian-upper",
        "_type";
        "algorithmic";
    }
    "armnlow";
    {
        "_rules";
        "armenian-lower",
        "_type";
        "algorithmic";
    }
    "bali";
    {
        "_digits";
        "᭐᭄ᭅᭆᭇᭈᭉᭊᭋᭌ᭍᭎᭏",
        "_type";
        "numeric";
    }
    "beng";
    {
        "_digits";
        "୦୧୨୩୪୫୬୭୮୯",
        "_type";
        "numeric";
    }
    "bhks";
    {

```

```

        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "brah";
    {
        "_digits";
        "ᱠᱡᱢᱯᱤᱨᱫᱽᱜᱟᱲ",
        "_type";
        "numeric";
    }
    "cakm";
    {
        "_digits";
        "ᱵᱚᱠᱟᱨᱫᱽᱜᱟᱲ",
        "_type";
        "numeric";
    }
    "cham";
    {
        "_digits";
        "ᱠᱡᱢᱯᱤᱨᱫᱽᱜᱟᱲ",
        "_type";
        "numeric";
    }
    "cyr1";
    {
        "_rules";
        "cyrillic-lower",
        "_type";
        "algorithmic";
    }
    "deva";
    {
        "_digits";
        "ᱠᱡᱢᱯᱤᱨᱫᱽᱜᱟᱲ",
        "_type";
        "numeric";
    }
    "ethi";
    {
        "_rules";
        "ethiopic",
        "_type";
        "algorithmic";
    }
    "fullwide";
    {
        "_digits";
        "0 1 2 3 4 5 6 7 8 9",
        "_type";
        "numeric";
    }
    "geor";
    {

```

```

        "_rules";
        "georgian",
        "_type";
        "algorithmic";
    }
    "grek";
    {
        "_rules";
        "greek-upper",
        "_type";
        "algorithmic";
    }
    "greklow";
    {
        "_rules";
        "greek-lower",
        "_type";
        "algorithmic";
    }
    "gujr";
    {
        "_digits";
        "૦૧૨૩૪૫૬૭૮૯",
        "_type";
        "numeric";
    }
    "guru";
    {
        "_digits";
        "੦੧੨੩੪੫੬੭੮੯",
        "_type";
        "numeric";
    }
    "hanidays";
    {
        "_rules";
        "zh/SpelloutRules/spellout-numbering-days",
        "_type";
        "algorithmic";
    }
    "hanidec";
    {
        "_digits";
        "〇一二三四五六七八九",
        "_type";
        "numeric";
    }
    "hans";
    {
        "_rules";
        "zh/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "hansfin";
    {

```

```

    "_rules";
    "zh/SpelloutRules/spellout-cardinal-financial",
        "_type";
    "algorithmic";
}
"hant";
{
    "_rules";
    "zh_Hant/SpelloutRules/spellout-cardinal",
        "_type";
    "algorithmic";
}
"hantfin";
{
    "_rules";
    "zh_Hant/SpelloutRules/spellout-cardinal-financial",
        "_type";
    "algorithmic";
}
"hebr";
{
    "_rules";
    "hebrew",
        "_type";
    "algorithmic";
}
"hmng";
{
    "_digits";
    "□□□□□□□□□□",
        "_type";
    "numeric";
}
"java";
{
    "_digits";
    "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉᐊᐋᐌᐍᐎᐏᐐᐑᐒᐓᐔᐕᐖᐗᐘᐙᐚᐛᐜᐝᐞᐟᐠᐡᐢᐣᐤᐥᐦᐧᐨᐩᐪᐫᐬᐭᐮᐯᐰᐱᐲᐳᐴᐵᐶᐷᐸᐹᐺᐻᐼᐽᐾᐿ",
        "_type";
    "numeric";
}
"jpan";
{
    "_rules";
    "ja/SpelloutRules/spellout-cardinal",
        "_type";
    "algorithmic";
}
"jpanfin";
{
    "_rules";
    "ja/SpelloutRules/spellout-cardinal-financial",
        "_type";
    "algorithmic";
}
"kali";
{

```

```

        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "khmr";
    {
        "_digits";
        "០១២៣៤៥៦៧៨៩",
        "_type";
        "numeric";
    }
    "knda";
    {
        "_digits";
        "೦೧೨೩೪೫೬೭೮೯",
        "_type";
        "numeric";
    }
    "lana";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "lanatham";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "laoo";
    {
        "_digits";
        "໐໑໒໓໔໕໖໗໘໑",
        "_type";
        "numeric";
    }
    "latn";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "lepc";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "limb";
    {

```

```

        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "mathbold";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathdbl";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathmono";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathsanb";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathsans";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mlym";
    {
        "_digits";
        "൦൧൨൩൪൫൬൭൮൯",
        "_type";
        "numeric";
    }
    "modi";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "mong";
    {

```

```

        "_digits";
        "၀၇၇၇၇၇၇၇",
        "_type";
        "numeric";
    }
    "mroo";
    {
        "_digits";
        "၀၀၀၀၀၀၀၀၀၀",
        "_type";
        "numeric";
    }
    "mtei";
    {
        "_digits";
        "၀၄၈၈၈၈၈၈၈၈၈",
        "_type";
        "numeric";
    }
    "mymr";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "mymrshan";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈",
        "_type";
        "numeric";
    }
    "mymrtlng";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "newa";
    {
        "_digits";
        "၀၀၀၀၀၀၀၀၀၀",
        "_type";
        "numeric";
    }
    "nkoo";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
}

```

```

"olck";
{
    "_digits";
    "00123456789",
    "_type";
    "numeric";
}
"orya";
{
    "_digits";
    "0099887766",
    "_type";
    "numeric";
}
"osma";
{
    "_digits";
    "05678901234",
    "_type";
    "numeric";
}
"roman";
{
    "_rules";
    "roman-upper",
    "_type";
    "algorithmic";
}
"romanlow";
{
    "_rules";
    "roman-lower",
    "_type";
    "algorithmic";
}
"saur";
{
    "_digits";
    "0000000000",
    "_type";
    "numeric";
}
"shrd";
{
    "_digits";
    "0000000000",
    "_type";
    "numeric";
}
"sind";
{
    "_digits";
    "0000000000",
    "_type";
    "numeric";
}

```



```

"sinh";
{
  "_digits";
  "൧൨൩൪൫൬൭൮൯",
  "_type";
  "numeric";
}
"sora";
{
  "_digits";
  "0123456789",
  "_type";
  "numeric";
}
"sund";
{
  "_digits";
  "□□□□□□□□",
  "_type";
  "numeric";
}
"takr";
{
  "_digits";
  "□□□□□□□□",
  "_type";
  "numeric";
}
"taluk";
{
  "_digits";
  "0123456789",
  "_type";
  "numeric";
}
"tamil";
{
  "_rules";
  "tamil",
  "_type";
  "algorithmic";
}
"tamildec";
{
  "_digits";
  "0௧௨௩௪௫௬௭௮௯",
  "_type";
  "numeric";
}
"telu";
{
  "_digits";
  "0౧౨౩౪౫౬౭౮౯",
  "_type";
  "numeric";
}

```

```

        "thai";
        {
            "_digits";
            "๐๑๒๓๔๕๖๗๘๙",
            "_type";
            "numeric";
        }
        "tibth";
        {
            "_digits";
            "༠༡༢༣༤༥༦༧༨༩",
            "_type";
            "numeric";
        }
        "tirh";
        {
            "_digits";
            "□□□□□□□□□□",
            "_type";
            "numeric";
        }
        "vaih";
        {
            "_digits";
            "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍᱎᱏ᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱹᱺᱻᱼᱽ᱾᱿",
            "_type";
            "numeric";
        }
        "wara";
        {
            "_digits";
            "□□□□□□□□□□",
            "_type";
            "numeric";
        }
    }
}

```

NUMBERS.JSON

```

{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31"
        },
        "language": "de"
      },
      "numbers": {
        "defaultNumberingSystem": "latn",
        "otherNumberingSystems": {
          "native": "latn"
        }
      }
    }
  }
}

```

```

    "minimumGroupingDigits": "1",
    "symbols-numberSystem-latn": {
      "decimal": ",",
      "group": ".",
      "list": ";",
      "percentSign": "%",
      "plusSign": "+",
      "minusSign": "-",
      "exponential": "E",
      "superscriptingExponent": "°",
      "perMille": "‰",
      "infinity": "∞",
      "nan": "NaN",
      "timeSeparator": ":"
    },
    "decimalFormats-numberSystem-latn": {
      "standard": "#,##0.###",
      "long": {
        "decimalFormat": {
          "1000-count-one": "0 Tausend",
          "1000-count-other": "0 Tausend",
          "10000-count-one": "00 Tausend",
          "10000-count-other": "00 Tausend",
          "100000-count-one": "000 Tausend",
          "100000-count-other": "000 Tausend",
          "1000000-count-one": "0 Million",
          "1000000-count-other": "0 Millionen",
          "10000000-count-one": "00 Millionen",
          "10000000-count-other": "00 Millionen",
          "100000000-count-one": "000 Millionen",
          "100000000-count-other": "000 Millionen",
          "1000000000-count-one": "0 Milliarde",
          "1000000000-count-other": "0 Milliarden",
          "10000000000-count-one": "00 Milliarden",
          "10000000000-count-other": "00 Milliarden",
          "100000000000-count-one": "000 Milliarden",
          "100000000000-count-other": "000 Milliarden",
          "1000000000000-count-one": "0 Billion",
          "1000000000000-count-other": "0 Billionen",
          "10000000000000-count-one": "00 Billionen",
          "10000000000000-count-other": "00 Billionen",
          "100000000000000-count-one": "000 Billionen",
          "100000000000000-count-other": "000 Billionen"
        }
      },
      "short": {
        "decimalFormat": {
          "1000-count-one": "0",
          "1000-count-other": "0",
          "10000-count-one": "0",
          "10000-count-other": "0",
          "100000-count-one": "0",
          "100000-count-other": "0",
          "1000000-count-one": "0 Mio'.'",
          "1000000-count-other": "0 Mio'.'",
          "10000000-count-one": "00 Mio'.'",
          "10000000-count-other": "00 Mio'.'"
        }
      }
    }
  }

```

```

        "1000000000-count-one": "000 Mio'.'",
        "1000000000-count-other": "000 Mio'.'",
        "1000000000-count-one": "0 Mrd'.'",
        "1000000000-count-other": "0 Mrd'.'",
        "10000000000-count-one": "00 Mrd'.'",
        "10000000000-count-other": "00 Mrd'.'",
        "100000000000-count-one": "000 Mrd'.'",
        "100000000000-count-other": "000 Mrd'.'",
        "1000000000000-count-one": "0 Bio'.'",
        "1000000000000-count-other": "0 Bio'.'",
        "10000000000000-count-one": "00 Bio'.'",
        "10000000000000-count-other": "00 Bio'.'",
        "100000000000000-count-one": "000 Bio'.'",
        "100000000000000-count-other": "000 Bio'.'"
    }
  },
  },
  "scientificFormats-numberSystem-latn": {
    "standard": "#E0"
  },
  },
  "percentFormats-numberSystem-latn": {
    "standard": "#,##0 %"
  },
  },
  "currencyFormats-numberSystem-latn": {
    "currencySpacing": {
      "beforeCurrency": {
        "currencyMatch": "[:^S:]",
        "surroundingMatch": "[:digit:]",
        "insertBetween": " "
      },
      "afterCurrency": {
        "currencyMatch": "[:^S:]",
        "surroundingMatch": "[:digit:]",
        "insertBetween": " "
      }
    },
    "standard": "#,##0.00 ¤",
    "accounting": "#,##0.00 ¤",
    "short": {
      "standard": {
        "1000-count-one": "0 Tsd'.' ¤",
        "1000-count-other": "0 Tsd'.' ¤",
        "10000-count-one": "00 Tsd'.' ¤",
        "10000-count-other": "00 Tsd'.' ¤",
        "100000-count-one": "000 Tsd'.' ¤",
        "100000-count-other": "000 Tsd'.' ¤",
        "1000000-count-one": "0 Mio'.' ¤",
        "1000000-count-other": "0 Mio'.' ¤",
        "10000000-count-one": "00 Mio'.' ¤",
        "10000000-count-other": "00 Mio'.' ¤",
        "100000000-count-one": "000 Mio'.' ¤",
        "100000000-count-other": "000 Mio'.' ¤",
        "1000000000-count-one": "0 Mrd'.' ¤",
        "1000000000-count-other": "0 Mrd'.' ¤",
        "10000000000-count-one": "00 Mrd'.' ¤",
        "10000000000-count-other": "00 Mrd'.' ¤",
        "100000000000-count-one": "000 Mrd'.' ¤",
        "100000000000-count-other": "000 Mrd'.' ¤",

```

```

        "100000000000-count-other": "000 Mrd'.' ¤",
        "100000000000-count-one": "0 Bio'.' ¤",
        "100000000000-count-other": "0 Bio'.' ¤",
        "100000000000-count-one": "00 Bio'.' ¤",
        "100000000000-count-other": "00 Bio'.' ¤",
        "100000000000-count-one": "000 Bio'.' ¤",
        "100000000000-count-other": "000 Bio'.' ¤"
    },
    },
    "unitPattern-count-one": "{0} {1}",
    "unitPattern-count-other": "{0} {1}"
},
"miscPatterns-numberSystem-latn": {
    "atLeast": "{0}+",
    "range": "{0}-{1}"
},
"minimalPairs": {
    "pluralMinimalPairs": "{0} Tag",
    "pluralMinimalPairs": "{0} Tage",
    "other": "{0}. Abzweigung nach rechts nehmen"
}
}
}
}
}
}

```

NUMBERS.JSX

```

{
    "main";
    {
        "de";
        {
            "identity";
            {
                "version";
                {
                    "_number";
                    "$Revision: 13259 $",
                    "_cldrVersion";
                    "31";
                }
                "language";
                "de";
            }
            "numbers";
            {
                "defaultNumberingSystem";
                "latn",
                "otherNumberingSystems";
                {
                    "native";
                    "latn";
                }
                "minimumGroupingDigits";
                "1",
            }
        }
    }
}

```

```

        "symbols-numberSystem-latn";
    {
        "decimal";
        ",",
        "group";
        ".",
        "list";
        ";",
        "percentSign";
        "%",
        "plusSign";
        "+",
        "minusSign";
        "-",
        "exponential";
        "E",
        "superscriptingExponent";
        ".",
        "perMille";
        "‰",
        "infinity";
        "∞",
        "nan";
        "NaN",
        "timeSeparator";
        ":";
    }
    "decimalFormats-numberSystem-latn";
    {
        "standard";
        "#,##0.###",
        "long";
        {
            "decimalFormat";
            {
                "1000-count-one";
                "0 Tausend",
                "1000-count-other";
                "0 Tausend",
                "10000-count-one";
                "00 Tausend",
                "10000-count-other";
                "00 Tausend",
                "100000-count-one";
                "000 Tausend",
                "100000-count-other";
                "000 Tausend",
                "1000000-count-one";
                "0 Million",
                "1000000-count-other";
                "0 Millionen",
                "10000000-count-one";
                "00 Millionen",
                "10000000-count-other";
                "00 Millionen",
                "100000000-count-one";
                "000 Millionen",
            }
        }
    }

```

```

        "100000000-count-other";
        "000 Millionen",
        "1000000000-count-one";
        "0 Milliarde",
        "1000000000-count-other";
        "0 Milliarden",
        "10000000000-count-one";
        "00 Milliarden",
        "10000000000-count-other";
        "00 Milliarden",
        "100000000000-count-one";
        "000 Milliarden",
        "100000000000-count-other";
        "000 Milliarden",
        "1000000000000-count-one";
        "0 Billion",
        "1000000000000-count-other";
        "0 Billionen",
        "10000000000000-count-one";
        "00 Billionen",
        "10000000000000-count-other";
        "00 Billionen",
        "100000000000000-count-one";
        "000 Billionen",
        "100000000000000-count-other";
        "000 Billionen";
    }
}
"short";
{
    "decimalFormat";
    {
        "1000-count-one";
        "0",
        "1000-count-other";
        "0",
        "10000-count-one";
        "0",
        "10000-count-other";
        "0",
        "100000-count-one";
        "0",
        "100000-count-other";
        "0",
        "1000000-count-one";
        "0 Mio'.'",
        "1000000-count-other";
        "0 Mio'.'",
        "10000000-count-one";
        "00 Mio'.'",
        "10000000-count-other";
        "00 Mio'.'",
        "100000000-count-one";
        "000 Mio'.'",
        "100000000-count-other";
        "000 Mio'.'",
        "1000000000-count-one";
    }
}

```

```

        "0 Mrd'.'",
        "10000000000-count-other";
        "0 Mrd'.'",
        "10000000000-count-one";
        "00 Mrd'.'",
        "10000000000-count-other";
        "00 Mrd'.'",
        "10000000000-count-one";
        "000 Mrd'.'",
        "10000000000-count-other";
        "000 Mrd'.'",
        "10000000000-count-one";
        "0 Bio'.'",
        "100000000000-count-other";
        "0 Bio'.'",
        "1000000000000-count-one";
        "00 Bio'.'",
        "1000000000000-count-other";
        "00 Bio'.'",
        "10000000000000-count-one";
        "000 Bio'.'",
        "100000000000000-count-other";
        "000 Bio'.'";
    }
}
}
"scientificFormats-numberSystem-latn";
{
    "standard";
    "#E0";
}
"percentFormats-numberSystem-latn";
{
    "standard";
    "#,##0 %";
}
"currencyFormats-numberSystem-latn";
{
    "currencySpacing";
    {
        "beforeCurrency";
        {
            "currencyMatch";
            "[:^S:]",
            "surroundingMatch";
            "[:digit:]",
            "insertBetween";
            " ";
        }
        "afterCurrency";
        {
            "currencyMatch";
            "[:^S:]",
            "surroundingMatch";
            "[:digit:]",
            "insertBetween";
            " ";
        }
    }
}

```



```

    }
  }
  "standard";
  "#,##0.00 ¤",
    "accounting";
  "#,##0.00 ¤",
    "short";
  {
    "standard";
    {
      "1000-count-one";
      "0 Tsd'.' ¤",
        "1000-count-other";
      "0 Tsd'.' ¤",
        "10000-count-one";
      "00 Tsd'.' ¤",
        "10000-count-other";
      "00 Tsd'.' ¤",
        "100000-count-one";
      "000 Tsd'.' ¤",
        "100000-count-other";
      "000 Tsd'.' ¤",
        "1000000-count-one";
      "0 Mio'.' ¤",
        "1000000-count-other";
      "0 Mio'.' ¤",
        "10000000-count-one";
      "00 Mio'.' ¤",
        "10000000-count-other";
      "00 Mio'.' ¤",
        "100000000-count-one";
      "000 Mio'.' ¤",
        "100000000-count-other";
      "000 Mio'.' ¤",
        "1000000000-count-one";
      "0 Mrd'.' ¤",
        "1000000000-count-other";
      "0 Mrd'.' ¤",
        "10000000000-count-one";
      "00 Mrd'.' ¤",
        "10000000000-count-other";
      "00 Mrd'.' ¤",
        "100000000000-count-one";
      "000 Mrd'.' ¤",
        "100000000000-count-other";
      "000 Mrd'.' ¤",
        "1000000000000-count-one";
      "0 Bio'.' ¤",
        "1000000000000-count-other";
      "0 Bio'.' ¤",
        "10000000000000-count-one";
      "00 Bio'.' ¤",
        "10000000000000-count-other";
      "00 Bio'.' ¤",
        "100000000000000-count-one";
      "000 Bio'.' ¤",
        "100000000000000-count-other";
    }
  }

```

```

        "000 Bio'.' α";
    }
}
    "unitPattern-count-one";
    "{0} {1}",
        "unitPattern-count-other";
    "{0} {1}";
}
"miscPatterns-numberSystem-latn";
{
    "atLeast";
    "{0}+",
        "range";
    "{0}-{1}";
}
"minimalPairs";
{
    "pluralMinimalPairs";
    "{0} Tag",
        "pluralMinimalPairs";
    "{0} Tage",
        "other";
    "{0}. Abzweigung nach rechts nehmen";
}
}
}
}

```

TIMEZONENAMES.JSON

```
{
  "main": {
    "de": {
      "identity": {
        "version": {
          "_number": "$Revision: 12879 $",
          "_cldrVersion": "30.0.3"
        },
        "language": "de"
      },
      "dates": {
        "timeZoneNames": {
          "hourFormat": "+HH:mm;-HH:mm",
          "gmtFormat": "GMT{0}",
          "gmtZeroFormat": "GMT",
          "regionFormat": "{0} Zeit",
          "regionFormat-type-daylight": "{0} Sommerzeit",
          "regionFormat-type-standard": "{0} Normalzeit",
          "fallbackFormat": "{1} ({0})",
          "zone": {
            "America": {
              "Adak": {
                "exemplarCity": "Adak"
              },
              "Anchorage": {
```

```
    "exemplarCity": "Anchorage"
  },
  "Anguilla": {
    "exemplarCity": "Anguilla"
  },
  "Antigua": {
    "exemplarCity": "Antigua"
  },
  "Araguaina": {
    "exemplarCity": "Araguaina"
  },
  "Argentina": {
    "Rio_Gallegos": {
      "exemplarCity": "Rio Gallegos"
    },
    "San_Juan": {
      "exemplarCity": "San Juan"
    },
    "Ushuaia": {
      "exemplarCity": "Ushuaia"
    },
    "La_Rioja": {
      "exemplarCity": "La Rioja"
    },
    "San_Luis": {
      "exemplarCity": "San Luis"
    },
    "Salta": {
      "exemplarCity": "Salta"
    },
    "Tucuman": {
      "exemplarCity": "Tucuman"
    }
  },
  "Aruba": {
    "exemplarCity": "Aruba"
  },
  "Asuncion": {
    "exemplarCity": "Asunción"
  },
  "Bahia": {
    "exemplarCity": "Bahia"
  },
  "Bahia_Banderas": {
    "exemplarCity": "Bahia Banderas"
  },
  "Barbados": {
    "exemplarCity": "Barbados"
  },
  "Belem": {
    "exemplarCity": "Belem"
  },
  "Belize": {
    "exemplarCity": "Belize"
  },
  "Blanc-Sablon": {
    "exemplarCity": "Blanc-Sablon"
  }
```

```
    },  
    "Boa_Vista": {  
      "exemplarCity": "Boa Vista"  
    },  
    "Bogota": {  
      "exemplarCity": "Bogotá"  
    },  
    "Boise": {  
      "exemplarCity": "Boise"  
    },  
    "Buenos_Aires": {  
      "exemplarCity": "Buenos Aires"  
    },  
    "Cambridge_Bay": {  
      "exemplarCity": "Cambridge Bay"  
    },  
    "Campo_Grande": {  
      "exemplarCity": "Campo Grande"  
    },  
    "Cancun": {  
      "exemplarCity": "Cancún"  
    },  
    "Caracas": {  
      "exemplarCity": "Caracas"  
    },  
    "Catamarca": {  
      "exemplarCity": "Catamarca"  
    },  
    "Cayenne": {  
      "exemplarCity": "Cayenne"  
    },  
    "Cayman": {  
      "exemplarCity": "Kaimaninseln"  
    },  
    "Chicago": {  
      "exemplarCity": "Chicago"  
    },  
    "Chihuahua": {  
      "exemplarCity": "Chihuahua"  
    },  
    "Coral_Harbour": {  
      "exemplarCity": "Atikokan"  
    },  
    "Cordoba": {  
      "exemplarCity": "Córdoba"  
    },  
    "Costa_Rica": {  
      "exemplarCity": "Costa Rica"  
    },  
    "Creston": {  
      "exemplarCity": "Creston"  
    },  
    "Cuiaba": {  
      "exemplarCity": "Cuiaba"  
    },  
    "Curacao": {  
      "exemplarCity": "Curaçao"
```

```
    },
    "Danmarkshavn": {
      "exemplarCity": "Danmarkshavn"
    },
    "Dawson": {
      "exemplarCity": "Dawson"
    },
    "Dawson_Creek": {
      "exemplarCity": "Dawson Creek"
    },
    "Denver": {
      "exemplarCity": "Denver"
    },
    "Detroit": {
      "exemplarCity": "Detroit"
    },
    "Dominica": {
      "exemplarCity": "Dominica"
    },
    "Edmonton": {
      "exemplarCity": "Edmonton"
    },
    "Eirunepe": {
      "exemplarCity": "Eirunepe"
    },
    "El_Salvador": {
      "exemplarCity": "El Salvador"
    },
    "Fort_Nelson": {
      "exemplarCity": "Fort Nelson"
    },
    "Fortaleza": {
      "exemplarCity": "Fortaleza"
    },
    "Glace_Bay": {
      "exemplarCity": "Glace Bay"
    },
    "Godthab": {
      "exemplarCity": "Nuuk"
    },
    "Goose_Bay": {
      "exemplarCity": "Goose Bay"
    },
    "Grand_Turk": {
      "exemplarCity": "Grand Turk"
    },
    "Grenada": {
      "exemplarCity": "Grenada"
    },
    "Guadeloupe": {
      "exemplarCity": "Guadeloupe"
    },
    "Guatemala": {
      "exemplarCity": "Guatemala"
    },
    "Guayaquil": {
      "exemplarCity": "Guayaquil"
    }
```

```
    },  
    "Guyana": {  
      "exemplarCity": "Guyana"  
    },  
    "Halifax": {  
      "exemplarCity": "Halifax"  
    },  
    "Havana": {  
      "exemplarCity": "Havanna"  
    },  
    "Hermosillo": {  
      "exemplarCity": "Hermosillo"  
    },  
    "Indiana": {  
      "Vincennes": {  
        "exemplarCity": "Vincennes, Indiana"  
      },  
      "Petersburg": {  
        "exemplarCity": "Petersburg, Indiana"  
      },  
      "Tell_City": {  
        "exemplarCity": "Tell City, Indiana"  
      },  
      "Knox": {  
        "exemplarCity": "Knox, Indiana"  
      },  
      "Winamac": {  
        "exemplarCity": "Winamac, Indiana"  
      },  
      "Marengo": {  
        "exemplarCity": "Marengo, Indiana"  
      },  
      "Vevay": {  
        "exemplarCity": "Vevay, Indiana"  
      }  
    },  
    "Indianapolis": {  
      "exemplarCity": "Indianapolis"  
    },  
    "Inuvik": {  
      "exemplarCity": "Inuvik"  
    },  
    "Iqaluit": {  
      "exemplarCity": "Iqaluit"  
    },  
    "Jamaica": {  
      "exemplarCity": "Jamaika"  
    },  
    "Jujuy": {  
      "exemplarCity": "Jujuy"  
    },  
    "Juneau": {  
      "exemplarCity": "Juneau"  
    },  
    "Kentucky": {  
      "Monticello": {  
        "exemplarCity": "Monticello, Kentucky"  
      }  
    }  
  }  
}
```

```
    },
    "Kralendijk": {
      "exemplarCity": "Kralendijk"
    },
    "La_Paz": {
      "exemplarCity": "La Paz"
    },
    "Lima": {
      "exemplarCity": "Lima"
    },
    "Los_Angeles": {
      "exemplarCity": "Los Angeles"
    },
    "Louisville": {
      "exemplarCity": "Louisville"
    },
    "Lower_Princes": {
      "exemplarCity": "Lower Prince's Quarter"
    },
    "Maceio": {
      "exemplarCity": "Maceio"
    },
    "Managua": {
      "exemplarCity": "Managua"
    },
    "Manaus": {
      "exemplarCity": "Manaus"
    },
    "Marigot": {
      "exemplarCity": "Marigot"
    },
    "Martinique": {
      "exemplarCity": "Martinique"
    },
    "Matamoros": {
      "exemplarCity": "Matamoros"
    },
    "Mazatlan": {
      "exemplarCity": "Mazatlan"
    },
    "Mendoza": {
      "exemplarCity": "Mendoza"
    },
    "Menominee": {
      "exemplarCity": "Menominee"
    },
    "Merida": {
      "exemplarCity": "Merida"
    },
    "Metlakatla": {
      "exemplarCity": "Metlakatla"
    },
    "Mexico_City": {
      "exemplarCity": "Mexiko-Stadt"
    },
    "Miquelon": {
```

```
        "exemplarCity": "Miquelon"
      },
      "Moncton": {
        "exemplarCity": "Moncton"
      },
      "Monterrey": {
        "exemplarCity": "Monterrey"
      },
      "Montevideo": {
        "exemplarCity": "Montevideo"
      },
      "Montserrat": {
        "exemplarCity": "Montserrat"
      },
      "Nassau": {
        "exemplarCity": "Nassau"
      },
      "New_York": {
        "exemplarCity": "New York"
      },
      "Nipigon": {
        "exemplarCity": "Nipigon"
      },
      "Nome": {
        "exemplarCity": "Nome"
      },
      "Noronha": {
        "exemplarCity": "Noronha"
      },
      "North_Dakota": {
        "Beulah": {
          "exemplarCity": "Beulah, North Dakota"
        },
        "New_Salem": {
          "exemplarCity": "New Salem, North Dakota"
        },
        "Center": {
          "exemplarCity": "Center, North Dakota"
        }
      },
      "Ojinaga": {
        "exemplarCity": "Ojinaga"
      },
      "Panama": {
        "exemplarCity": "Panama"
      },
      "Pangnirtung": {
        "exemplarCity": "Pangnirtung"
      },
      "Paramaribo": {
        "exemplarCity": "Paramaribo"
      },
      "Phoenix": {
        "exemplarCity": "Phoenix"
      },
      "Port-au-Prince": {
        "exemplarCity": "Port-au-Prince"
      }
    }
  }
}
```



```
    },
    "Port_of_Spain": {
      "exemplarCity": "Port of Spain"
    },
    "Porto_Velho": {
      "exemplarCity": "Porto Velho"
    },
    "Puerto_Rico": {
      "exemplarCity": "Puerto Rico"
    },
    "Rainy_River": {
      "exemplarCity": "Rainy River"
    },
    "Rankin_Inlet": {
      "exemplarCity": "Rankin Inlet"
    },
    "Recife": {
      "exemplarCity": "Recife"
    },
    "Regina": {
      "exemplarCity": "Regina"
    },
    "Resolute": {
      "exemplarCity": "Resolute"
    },
    "Rio_Branco": {
      "exemplarCity": "Rio Branco"
    },
    "Santa_Isabel": {
      "exemplarCity": "Santa Isabel"
    },
    "Santarem": {
      "exemplarCity": "Santarem"
    },
    "Santiago": {
      "exemplarCity": "Santiago"
    },
    "Santo_Domingo": {
      "exemplarCity": "Santo Domingo"
    },
    "Sao_Paulo": {
      "exemplarCity": "São Paulo"
    },
    "Scoresbysund": {
      "exemplarCity": "Ittoqqortoormiit"
    },
    "Sitka": {
      "exemplarCity": "Sitka"
    },
    "St_Barthelemy": {
      "exemplarCity": "Saint-Barthélemy"
    },
    "St_Johns": {
      "exemplarCity": "St. John's"
    },
    "St_Kitts": {
      "exemplarCity": "St. Kitts"
    }
  }
```

```
    },
    "St_Lucia": {
      "exemplarCity": "St. Lucia"
    },
    "St_Thomas": {
      "exemplarCity": "St. Thomas"
    },
    "St_Vincent": {
      "exemplarCity": "St. Vincent"
    },
    "Swift_Current": {
      "exemplarCity": "Swift Current"
    },
    "Tegucigalpa": {
      "exemplarCity": "Tegucigalpa"
    },
    "Thule": {
      "exemplarCity": "Thule"
    },
    "Thunder_Bay": {
      "exemplarCity": "Thunder Bay"
    },
    "Tijuana": {
      "exemplarCity": "Tijuana"
    },
    "Toronto": {
      "exemplarCity": "Toronto"
    },
    "Tortola": {
      "exemplarCity": "Tortola"
    },
    "Vancouver": {
      "exemplarCity": "Vancouver"
    },
    "Whitehorse": {
      "exemplarCity": "Whitehorse"
    },
    "Winnipeg": {
      "exemplarCity": "Winnipeg"
    },
    "Yakutat": {
      "exemplarCity": "Yakutat"
    },
    "Yellowknife": {
      "exemplarCity": "Yellowknife"
    }
  },
  "Atlantic": {
    "Azores": {
      "exemplarCity": "Azoren"
    },
    "Bermuda": {
      "exemplarCity": "Bermudas"
    },
    "Canary": {
      "exemplarCity": "Kanaren"
    }
  },
}
```

```
"Cape_Verde": {
  "exemplarCity": "Cabo Verde"
},
"Faeroe": {
  "exemplarCity": "Färöer"
},
"Madeira": {
  "exemplarCity": "Madeira"
},
"Reykjavik": {
  "exemplarCity": "Reykjavík"
},
"South_Georgia": {
  "exemplarCity": "Südgeorgien"
},
"St_Helena": {
  "exemplarCity": "St. Helena"
},
"Stanley": {
  "exemplarCity": "Stanley"
}
},
"Europe": {
  "Amsterdam": {
    "exemplarCity": "Amsterdam"
  },
  "Andorra": {
    "exemplarCity": "Andorra"
  },
  "Astrakhan": {
    "exemplarCity": "Astrachan"
  },
  "Athens": {
    "exemplarCity": "Athen"
  },
  "Belgrade": {
    "exemplarCity": "Belgrad"
  },
  "Berlin": {
    "exemplarCity": "Berlin"
  },
  "Bratislava": {
    "exemplarCity": "Bratislava"
  },
  "Brussels": {
    "exemplarCity": "Brüssel"
  },
  "Bucharest": {
    "exemplarCity": "Bukarest"
  },
  "Budapest": {
    "exemplarCity": "Budapest"
  },
  "Busingen": {
    "exemplarCity": "Büsingen"
  },
  "Chisinau": {
```

```
    "exemplarCity": "Kischinau"
  },
  "Copenhagen": {
    "exemplarCity": "Kopenhagen"
  },
  "Dublin": {
    "long": {
      "daylight": "Irische Sommerzeit"
    },
    "exemplarCity": "Dublin"
  },
  "Gibraltar": {
    "exemplarCity": "Gibraltar"
  },
  "Guernsey": {
    "exemplarCity": "Guernsey"
  },
  "Helsinki": {
    "exemplarCity": "Helsinki"
  },
  "Isle_of_Man": {
    "exemplarCity": "Isle of Man"
  },
  "Istanbul": {
    "exemplarCity": "Istanbul"
  },
  "Jersey": {
    "exemplarCity": "Jersey"
  },
  "Kaliningrad": {
    "exemplarCity": "Kaliningrad"
  },
  "Kiev": {
    "exemplarCity": "Kiew"
  },
  "Kirov": {
    "exemplarCity": "Kirow"
  },
  "Lisbon": {
    "exemplarCity": "Lissabon"
  },
  "Ljubljana": {
    "exemplarCity": "Ljubljana"
  },
  "London": {
    "long": {
      "daylight": "Britische Sommerzeit"
    },
    "exemplarCity": "London"
  },
  "Luxembourg": {
    "exemplarCity": "Luxemburg"
  },
  "Madrid": {
    "exemplarCity": "Madrid"
  },
  "Malta": {
```

```
        "exemplarCity": "Malta"
      },
      "Mariehamn": {
        "exemplarCity": "Mariehamn"
      },
      "Minsk": {
        "exemplarCity": "Minsk"
      },
      "Monaco": {
        "exemplarCity": "Monaco"
      },
      "Moscow": {
        "exemplarCity": "Moskau"
      },
      "Oslo": {
        "exemplarCity": "Oslo"
      },
      "Paris": {
        "exemplarCity": "Paris"
      },
      "Podgorica": {
        "exemplarCity": "Podgorica"
      },
      "Prague": {
        "exemplarCity": "Prag"
      },
      "Riga": {
        "exemplarCity": "Riga"
      },
      "Rome": {
        "exemplarCity": "Rom"
      },
      "Samara": {
        "exemplarCity": "Samara"
      },
      "San_Marino": {
        "exemplarCity": "San Marino"
      },
      "Sarajevo": {
        "exemplarCity": "Sarajevo"
      },
      "Simferopol": {
        "exemplarCity": "Simferopol"
      },
      "Skopje": {
        "exemplarCity": "Skopje"
      },
      "Sofia": {
        "exemplarCity": "Sofia"
      },
      "Stockholm": {
        "exemplarCity": "Stockholm"
      },
      "Tallinn": {
        "exemplarCity": "Tallinn"
      },
      "Tirane": {
```

```
        "exemplarCity": "Tirana"
      },
      "Ulyanovsk": {
        "exemplarCity": "Uljanowsk"
      },
      "Uzhgorod": {
        "exemplarCity": "Uschgorod"
      },
      "Vaduz": {
        "exemplarCity": "Vaduz"
      },
      "Vatican": {
        "exemplarCity": "Vatikan"
      },
      "Vienna": {
        "exemplarCity": "Wien"
      },
      "Vilnius": {
        "exemplarCity": "Vilnius"
      },
      "Volgograd": {
        "exemplarCity": "Wolgograd"
      },
      "Warsaw": {
        "exemplarCity": "Warschau"
      },
      "Zagreb": {
        "exemplarCity": "Zagreb"
      },
      "Zaporozhye": {
        "exemplarCity": "Saporischja"
      },
      "Zurich": {
        "exemplarCity": "Zürich"
      }
    },
    "Africa": {
      "Abidjan": {
        "exemplarCity": "Abidjan"
      },
      "Accra": {
        "exemplarCity": "Accra"
      },
      "Addis_Ababa": {
        "exemplarCity": "Addis Abeba"
      },
      "Algiers": {
        "exemplarCity": "Algier"
      },
      "Asmera": {
        "exemplarCity": "Asmara"
      },
      "Bamako": {
        "exemplarCity": "Bamako"
      },
      "Bangui": {
        "exemplarCity": "Bangui"
      }
    }
  }
}
```

```
    },  
    "Banjul": {  
      "exemplarCity": "Banjul"  
    },  
    "Bissau": {  
      "exemplarCity": "Bissau"  
    },  
    "Blantyre": {  
      "exemplarCity": "Blantyre"  
    },  
    "Brazzaville": {  
      "exemplarCity": "Brazzaville"  
    },  
    "Bujumbura": {  
      "exemplarCity": "Bujumbura"  
    },  
    "Cairo": {  
      "exemplarCity": "Kairo"  
    },  
    "Casablanca": {  
      "exemplarCity": "Casablanca"  
    },  
    "Ceuta": {  
      "exemplarCity": "Ceuta"  
    },  
    "Conakry": {  
      "exemplarCity": "Conakry"  
    },  
    "Dakar": {  
      "exemplarCity": "Dakar"  
    },  
    "Dar_es_Salaam": {  
      "exemplarCity": "Daressalam"  
    },  
    "Djibouti": {  
      "exemplarCity": "Dschibuti"  
    },  
    "Douala": {  
      "exemplarCity": "Douala"  
    },  
    "El_Aaiun": {  
      "exemplarCity": "El Aaiún"  
    },  
    "Freetown": {  
      "exemplarCity": "Freetown"  
    },  
    "Gaborone": {  
      "exemplarCity": "Gaborone"  
    },  
    "Harare": {  
      "exemplarCity": "Harare"  
    },  
    "Johannesburg": {  
      "exemplarCity": "Johannesburg"  
    },  
    "Juba": {  
      "exemplarCity": "Juba"
```

```
    },  
    "Kampala": {  
      "exemplarCity": "Kampala"  
    },  
    "Khartoum": {  
      "exemplarCity": "Khartum"  
    },  
    "Kigali": {  
      "exemplarCity": "Kigali"  
    },  
    "Kinshasa": {  
      "exemplarCity": "Kinshasa"  
    },  
    "Lagos": {  
      "exemplarCity": "Lagos"  
    },  
    "Libreville": {  
      "exemplarCity": "Libreville"  
    },  
    "Lome": {  
      "exemplarCity": "Lomé"  
    },  
    "Luanda": {  
      "exemplarCity": "Luanda"  
    },  
    "Lubumbashi": {  
      "exemplarCity": "Lubumbashi"  
    },  
    "Lusaka": {  
      "exemplarCity": "Lusaka"  
    },  
    "Malabo": {  
      "exemplarCity": "Malabo"  
    },  
    "Maputo": {  
      "exemplarCity": "Maputo"  
    },  
    "Maseru": {  
      "exemplarCity": "Maseru"  
    },  
    "Mbabane": {  
      "exemplarCity": "Mbabane"  
    },  
    "Mogadishu": {  
      "exemplarCity": "Mogadischu"  
    },  
    "Monrovia": {  
      "exemplarCity": "Monrovia"  
    },  
    "Nairobi": {  
      "exemplarCity": "Nairobi"  
    },  
    "Ndjamena": {  
      "exemplarCity": "N'Djamena"  
    },  
    "Niamey": {  
      "exemplarCity": "Niamey"
```



```
    },  
    "Nouakchott": {  
      "exemplarCity": "Nouakchott"  
    },  
    "Ouagadougou": {  
      "exemplarCity": "Ouagadougou"  
    },  
    "Porto-Novo": {  
      "exemplarCity": "Porto Novo"  
    },  
    "Sao_Tome": {  
      "exemplarCity": "São Tomé"  
    },  
    "Tripoli": {  
      "exemplarCity": "Tripolis"  
    },  
    "Tunis": {  
      "exemplarCity": "Tunis"  
    },  
    "Windhoek": {  
      "exemplarCity": "Windhoek"  
    }  
  },  
  "Asia": {  
    "Aden": {  
      "exemplarCity": "Aden"  
    },  
    "Almaty": {  
      "exemplarCity": "Almaty"  
    },  
    "Amman": {  
      "exemplarCity": "Amman"  
    },  
    "Anadyr": {  
      "exemplarCity": "Anadyr"  
    },  
    "Aqtau": {  
      "exemplarCity": "Aqtau"  
    },  
    "Aqtobe": {  
      "exemplarCity": "Aktobe"  
    },  
    "Ashgabat": {  
      "exemplarCity": "Aşgabat"  
    },  
    "Baghdad": {  
      "exemplarCity": "Bagdad"  
    },  
    "Bahrain": {  
      "exemplarCity": "Bahrain"  
    },  
    "Baku": {  
      "exemplarCity": "Baku"  
    },  
    "Bangkok": {  
      "exemplarCity": "Bangkok"  
    }  
  },  
}
```

```
"Barnaul": {
  "exemplarCity": "Barnaul"
},
"Beirut": {
  "exemplarCity": "Beirut"
},
"Bishkek": {
  "exemplarCity": "Bischkek"
},
"Brunei": {
  "exemplarCity": "Brunei"
},
"Calcutta": {
  "exemplarCity": "Kalkutta"
},
"Chita": {
  "exemplarCity": "Tschita"
},
"Choibalsan": {
  "exemplarCity": "Tschoibalsan"
},
"Colombo": {
  "exemplarCity": "Colombo"
},
"Damascus": {
  "exemplarCity": "Damaskus"
},
"Dhaka": {
  "exemplarCity": "Dhaka"
},
"Dili": {
  "exemplarCity": "Dili"
},
"Dubai": {
  "exemplarCity": "Dubai"
},
"Dushanbe": {
  "exemplarCity": "Duschanbe"
},
"Gaza": {
  "exemplarCity": "Gaza"
},
"Hebron": {
  "exemplarCity": "Hebron"
},
"Hong_Kong": {
  "exemplarCity": "Hongkong"
},
"Hovd": {
  "exemplarCity": "Chowd"
},
"Irkutsk": {
  "exemplarCity": "Irkutsk"
},
"Jakarta": {
  "exemplarCity": "Jakarta"
},
```

```
"Jayapura": {
  "exemplarCity": "Jayapura"
},
"Jerusalem": {
  "exemplarCity": "Jerusalem"
},
"Kabul": {
  "exemplarCity": "Kabul"
},
"Kamchatka": {
  "exemplarCity": "Kamtschatka"
},
"Karachi": {
  "exemplarCity": "Karatschi"
},
"Katmandu": {
  "exemplarCity": "Kathmandu"
},
"Khandyga": {
  "exemplarCity": "Chandyga"
},
"Krasnoyarsk": {
  "exemplarCity": "Krasnojarsk"
},
"Kuala_Lumpur": {
  "exemplarCity": "Kuala Lumpur"
},
"Kuching": {
  "exemplarCity": "Kuching"
},
"Kuwait": {
  "exemplarCity": "Kuwait"
},
"Macau": {
  "exemplarCity": "Macao"
},
"Magadan": {
  "exemplarCity": "Magadan"
},
"Makassar": {
  "exemplarCity": "Makassar"
},
"Manila": {
  "exemplarCity": "Manila"
},
"Muscat": {
  "exemplarCity": "Maskat"
},
"Nicosia": {
  "exemplarCity": "Nikosia"
},
"Novokuznetsk": {
  "exemplarCity": "Nowokuznetsk"
},
"Novosibirsk": {
  "exemplarCity": "Nowosibirsk"
},
}
```

```
"Omsk": {
  "exemplarCity": "Omsk"
},
"Oral": {
  "exemplarCity": "Oral"
},
"Phnom_Penh": {
  "exemplarCity": "Phnom Penh"
},
"Pontianak": {
  "exemplarCity": "Pontianak"
},
"Pyongyang": {
  "exemplarCity": "Pjöngjang"
},
"Qatar": {
  "exemplarCity": "Katar"
},
"Qyzylorda": {
  "exemplarCity": "Qysylorda"
},
"Rangoon": {
  "exemplarCity": "Rangun"
},
"Riyadh": {
  "exemplarCity": "Riad"
},
"Saigon": {
  "exemplarCity": "Ho-Chi-Minh-Stadt"
},
"Sakhalin": {
  "exemplarCity": "Sachalin"
},
"Samarkand": {
  "exemplarCity": "Samarkand"
},
"Seoul": {
  "exemplarCity": "Seoul"
},
"Shanghai": {
  "exemplarCity": "Shanghai"
},
"Singapore": {
  "exemplarCity": "Singapur"
},
"Srednekolymsk": {
  "exemplarCity": "Srednekolymsk"
},
"Taipei": {
  "exemplarCity": "Taipeh"
},
"Tashkent": {
  "exemplarCity": "Taschkent"
},
"Tbilisi": {
  "exemplarCity": "Tiflis"
},
}
```

```
"Tehran": {
  "exemplarCity": "Teheran"
},
"Thimphu": {
  "exemplarCity": "Thimphu"
},
"Tokyo": {
  "exemplarCity": "Tokio"
},
"Tomsk": {
  "exemplarCity": "Tomsk"
},
"Ulaanbaatar": {
  "exemplarCity": "Ulaanbaatar"
},
"Urumqi": {
  "exemplarCity": "Ürümqi"
},
"Ust-Nera": {
  "exemplarCity": "Ust-Nera"
},
"Vientiane": {
  "exemplarCity": "Vientiane"
},
"Vladivostok": {
  "exemplarCity": "Wladiwostok"
},
"Yakutsk": {
  "exemplarCity": "Jakutsk"
},
"Yekaterinburg": {
  "exemplarCity": "Jekaterinburg"
},
"Yerevan": {
  "exemplarCity": "Eriwan"
}
},
"Indian": {
  "Antananarivo": {
    "exemplarCity": "Antananarivo"
  },
  "Chagos": {
    "exemplarCity": "Chagos"
  },
  "Christmas": {
    "exemplarCity": "Weihnachtsinsel"
  },
  "Cocos": {
    "exemplarCity": "Cocos"
  },
  "Comoro": {
    "exemplarCity": "Komoren"
  },
  "Kerguelen": {
    "exemplarCity": "Kerguelen"
  },
  "Mahe": {
```

```
        "exemplarCity": "Mahe"
      },
      "Maldives": {
        "exemplarCity": "Malediven"
      },
      "Mauritius": {
        "exemplarCity": "Mauritius"
      },
      "Mayotte": {
        "exemplarCity": "Mayotte"
      },
      "Reunion": {
        "exemplarCity": "Réunion"
      }
    },
    "Australia": {
      "Adelaide": {
        "exemplarCity": "Adelaide"
      },
      "Brisbane": {
        "exemplarCity": "Brisbane"
      },
      "Broken_Hill": {
        "exemplarCity": "Broken Hill"
      },
      "Currie": {
        "exemplarCity": "Currie"
      },
      "Darwin": {
        "exemplarCity": "Darwin"
      },
      "Eucla": {
        "exemplarCity": "Eucla"
      },
      "Hobart": {
        "exemplarCity": "Hobart"
      },
      "Lindeman": {
        "exemplarCity": "Lindeman"
      },
      "Lord_Howe": {
        "exemplarCity": "Lord Howe"
      },
      "Melbourne": {
        "exemplarCity": "Melbourne"
      },
      "Perth": {
        "exemplarCity": "Perth"
      },
      "Sydney": {
        "exemplarCity": "Sydney"
      }
    },
    "Pacific": {
      "Apia": {
        "exemplarCity": "Apia"
      }
    }
  },
}
```

```
"Auckland": {
  "exemplarCity": "Auckland"
},
"Bougainville": {
  "exemplarCity": "Bougainville"
},
"Chatham": {
  "exemplarCity": "Chatham"
},
"Easter": {
  "exemplarCity": "Osterinsel"
},
"Efate": {
  "exemplarCity": "Efate"
},
"Enderbury": {
  "exemplarCity": "Enderbury"
},
"Fakaofo": {
  "exemplarCity": "Fakaofo"
},
"Fiji": {
  "exemplarCity": "Fidschi"
},
"Funafuti": {
  "exemplarCity": "Funafuti"
},
"Galapagos": {
  "exemplarCity": "Galapagos"
},
"Gambier": {
  "exemplarCity": "Gambier"
},
"Guadalcanal": {
  "exemplarCity": "Guadalcanal"
},
"Guam": {
  "exemplarCity": "Guam"
},
"Honolulu": {
  "exemplarCity": "Honolulu"
},
"Johnston": {
  "exemplarCity": "Johnston"
},
"Kiritimati": {
  "exemplarCity": "Kiritimati"
},
"Kosrae": {
  "exemplarCity": "Kosrae"
},
"Kwajalein": {
  "exemplarCity": "Kwajalein"
},
"Majuro": {
  "exemplarCity": "Majuro"
},
}
```

```
"Marquesas": {
  "exemplarCity": "Marquesas"
},
"Midway": {
  "exemplarCity": "Midway"
},
"Nauru": {
  "exemplarCity": "Nauru"
},
"Niue": {
  "exemplarCity": "Niue"
},
"Norfolk": {
  "exemplarCity": "Norfolk"
},
"Noumea": {
  "exemplarCity": "Noumea"
},
"Pago_Pago": {
  "exemplarCity": "Pago Pago"
},
"Palau": {
  "exemplarCity": "Palau"
},
"Pitcairn": {
  "exemplarCity": "Pitcairn"
},
"Ponape": {
  "exemplarCity": "Pohnpei"
},
"Port_Moresby": {
  "exemplarCity": "Port Moresby"
},
"Rarotonga": {
  "exemplarCity": "Rarotonga"
},
"Saipan": {
  "exemplarCity": "Saipan"
},
"Tahiti": {
  "exemplarCity": "Tahiti"
},
"Tarawa": {
  "exemplarCity": "Tarawa"
},
"Tongatapu": {
  "exemplarCity": "Tongatapu"
},
"Truk": {
  "exemplarCity": "Chuuk"
},
"Wake": {
  "exemplarCity": "Wake"
},
"Wallis": {
  "exemplarCity": "Wallis"
}
```



```
    },
    "Arctic": {
      "Longyearbyen": {
        "exemplarCity": "Longyearbyen"
      }
    },
    "Antarctica": {
      "Casey": {
        "exemplarCity": "Casey"
      },
      "Davis": {
        "exemplarCity": "Davis"
      },
      "DumontDUrville": {
        "exemplarCity": "Dumont d'Urville"
      },
      "Macquarie": {
        "exemplarCity": "Macquarie"
      },
      "Mawson": {
        "exemplarCity": "Mawson"
      },
      "McMurdo": {
        "exemplarCity": "McMurdo"
      },
      "Palmer": {
        "exemplarCity": "Palmer"
      },
      "Rothera": {
        "exemplarCity": "Rothera"
      },
      "Syowa": {
        "exemplarCity": "Syowa"
      },
      "Troll": {
        "exemplarCity": "Troll"
      },
      "Vostok": {
        "exemplarCity": "Wostok"
      }
    },
    "Etc": {
      "GMT": {
        "exemplarCity": "GMT"
      },
      "GMT1": {
        "exemplarCity": "GMT+1"
      },
      "GMT10": {
        "exemplarCity": "GMT+10"
      },
      "GMT11": {
        "exemplarCity": "GMT+11"
      },
      "GMT12": {
        "exemplarCity": "GMT+12"
      }
    }
  },
}
```

```
"GMT2": {
  "exemplarCity": "GMT+2"
},
"GMT3": {
  "exemplarCity": "GMT+3"
},
"GMT4": {
  "exemplarCity": "GMT+4"
},
"GMT5": {
  "exemplarCity": "GMT+5"
},
"GMT6": {
  "exemplarCity": "GMT+6"
},
"GMT7": {
  "exemplarCity": "GMT+7"
},
"GMT8": {
  "exemplarCity": "GMT+8"
},
"GMT9": {
  "exemplarCity": "GMT+9"
},
"GMT-1": {
  "exemplarCity": "GMT-1"
},
"GMT-10": {
  "exemplarCity": "GMT-10"
},
"GMT-11": {
  "exemplarCity": "GMT-11"
},
"GMT-12": {
  "exemplarCity": "GMT-12"
},
"GMT-13": {
  "exemplarCity": "GMT-13"
},
"GMT-14": {
  "exemplarCity": "GMT-14"
},
"GMT-2": {
  "exemplarCity": "GMT-2"
},
"GMT-3": {
  "exemplarCity": "GMT-3"
},
"GMT-4": {
  "exemplarCity": "GMT-4"
},
"GMT-5": {
  "exemplarCity": "GMT-5"
},
"GMT-6": {
  "exemplarCity": "GMT-6"
},
},
```

```

    "GMT-7": {
      "exemplarCity": "GMT-7"
    },
    "GMT-8": {
      "exemplarCity": "GMT-8"
    },
    "GMT-9": {
      "exemplarCity": "GMT-9"
    },
    "Unknown": {
      "exemplarCity": "Unbekannt"
    }
  },
  "metazone": {
    "Acre": {
      "long": {
        "generic": "Acre-Zeit",
        "standard": "Acre-Normalzeit",
        "daylight": "Acre-Sommerzeit"
      }
    },
    "Afghanistan": {
      "long": {
        "standard": "Afghanistan-Zeit"
      }
    },
    "Africa_Central": {
      "long": {
        "standard": "Zentralafrikanische Zeit"
      }
    },
    "Africa_Eastern": {
      "long": {
        "standard": "Ostafrikanische Zeit"
      }
    },
    "Africa_Southern": {
      "long": {
        "standard": "Südafrikanische Zeit"
      }
    },
    "Africa_Western": {
      "long": {
        "generic": "Westafrikanische Zeit",
        "standard": "Westafrikanische Normalzeit",
        "daylight": "Westafrikanische Sommerzeit"
      }
    },
    "Alaska": {
      "long": {
        "generic": "Alaska-Zeit",
        "standard": "Alaska-Normalzeit",
        "daylight": "Alaska-Sommerzeit"
      }
    },
    "Almaty": {

```

```

        "long": {
            "generic": "Almaty-Zeit",
            "standard": "Almaty-Normalzeit",
            "daylight": "Almaty-Sommerzeit"
        }
    },
    "Amazon": {
        "long": {
            "generic": "Amazonas-Zeit",
            "standard": "Amazonas-Normalzeit",
            "daylight": "Amazonas-Sommerzeit"
        }
    },
    "America_Central": {
        "long": {
            "generic": "Nordamerikanische Inlandzeit",
            "standard": "Nordamerikanische Inland-Normalzeit",
            "daylight": "Nordamerikanische Inland-Sommerzeit"
        }
    },
    "America_Eastern": {
        "long": {
            "generic": "Nordamerikanische Ostküstenzeit",
            "standard": "Nordamerikanische Ostküsten-Normalzeit",
            "daylight": "Nordamerikanische Ostküsten-Sommerzeit"
        }
    },
    "America_Mountain": {
        "long": {
            "generic": "Rocky-Mountain-Zeit",
            "standard": "Rocky Mountain-Normalzeit",
            "daylight": "Rocky-Mountain-Sommerzeit"
        }
    },
    "America_Pacific": {
        "long": {
            "generic": "Nordamerikanische Westküstenzeit",
            "standard": "Nordamerikanische Westküsten-Normalzeit",
            "daylight": "Nordamerikanische Westküsten-Sommerzeit"
        }
    },
    "Anadyr": {
        "long": {
            "generic": "Anadyr Zeit",
            "standard": "Anadyr Normalzeit",
            "daylight": "Anadyr Sommerzeit"
        }
    },
    "Apia": {
        "long": {
            "generic": "Apia-Zeit",
            "standard": "Apia-Normalzeit",
            "daylight": "Apia-Sommerzeit"
        }
    },
    "Aqtan": {
        "long": {

```

```

        "generic": "Aqtau-Zeit",
        "standard": "Aqtau-Normalzeit",
        "daylight": "Aqtau-Sommerzeit"
    },
},
"Aqtobe": {
    "long": {
        "generic": "Aqtöbe-Zeit",
        "standard": "Aqtöbe-Normalzeit",
        "daylight": "Aqtöbe-Sommerzeit"
    }
},
"Arabian": {
    "long": {
        "generic": "Arabische Zeit",
        "standard": "Arabische Normalzeit",
        "daylight": "Arabische Sommerzeit"
    }
},
"Argentina": {
    "long": {
        "generic": "Argentinische Zeit",
        "standard": "Argentinische Normalzeit",
        "daylight": "Argentinische Sommerzeit"
    }
},
"Argentina_Western": {
    "long": {
        "generic": "Westargentinische Zeit",
        "standard": "Westargentinische Normalzeit",
        "daylight": "Westargentinische Sommerzeit"
    }
},
"Armenia": {
    "long": {
        "generic": "Armenische Zeit",
        "standard": "Armenische Normalzeit",
        "daylight": "Armenische Sommerzeit"
    }
},
"Atlantic": {
    "long": {
        "generic": "Atlantik-Zeit",
        "standard": "Atlantik-Normalzeit",
        "daylight": "Atlantik-Sommerzeit"
    }
},
"Australia_Central": {
    "long": {
        "generic": "Zentralaustralische Zeit",
        "standard": "Zentralaustralische Normalzeit",
        "daylight": "Zentralaustralische Sommerzeit"
    }
},
"Australia_CentralWestern": {
    "long": {
        "generic": "Zentral-/Westaustralische Zeit",

```

```

        "standard": "Zentral-/Westaustralische Normalzeit",
        "daylight": "Zentral-/Westaustralische Sommerzeit"
    },
    "Australia_Eastern": {
        "long": {
            "generic": "Ostaustralische Zeit",
            "standard": "Ostaustralische Normalzeit",
            "daylight": "Ostaustralische Sommerzeit"
        }
    },
    "Australia_Western": {
        "long": {
            "generic": "Westaustralische Zeit",
            "standard": "Westaustralische Normalzeit",
            "daylight": "Westaustralische Sommerzeit"
        }
    },
    "Azerbaijan": {
        "long": {
            "generic": "Aserbaidsschanische Zeit",
            "standard": "Aserbeidschanische Normalzeit",
            "daylight": "Aserbaidsschanische Sommerzeit"
        }
    },
    "Azores": {
        "long": {
            "generic": "Azoren-Zeit",
            "standard": "Azoren-Normalzeit",
            "daylight": "Azoren-Sommerzeit"
        }
    },
    "Bangladesh": {
        "long": {
            "generic": "Bangladesch-Zeit",
            "standard": "Bangladesch-Normalzeit",
            "daylight": "Bangladesch-Sommerzeit"
        }
    },
    "Bhutan": {
        "long": {
            "standard": "Bhutan-Zeit"
        }
    },
    "Bolivia": {
        "long": {
            "standard": "Bolivianische Zeit"
        }
    },
    "Brasilia": {
        "long": {
            "generic": "Brasília-Zeit",
            "standard": "Brasília-Normalzeit",
            "daylight": "Brasília-Sommerzeit"
        }
    },
    "Brunei": {

```

```

        "long": {
            "standard": "Brunei-Zeit"
        }
    },
    "Cape_Verde": {
        "long": {
            "generic": "Cabo-Verde-Zeit",
            "standard": "Cabo-Verde-Normalzeit",
            "daylight": "Cabo-Verde-Sommerzeit"
        }
    },
    "Casey": {
        "long": {
            "standard": "Casey-Zeit"
        }
    },
    "Chamorro": {
        "long": {
            "standard": "Chamorro-Zeit"
        }
    },
    "Chatham": {
        "long": {
            "generic": "Chatham-Zeit",
            "standard": "Chatham-Normalzeit",
            "daylight": "Chatham-Sommerzeit"
        }
    },
    "Chile": {
        "long": {
            "generic": "Chilenische Zeit",
            "standard": "Chilenische Normalzeit",
            "daylight": "Chilenische Sommerzeit"
        }
    },
    "China": {
        "long": {
            "generic": "Chinesische Zeit",
            "standard": "Chinesische Normalzeit",
            "daylight": "Chinesische Sommerzeit"
        }
    },
    "Choibalsan": {
        "long": {
            "generic": "Tschoibalsan-Zeit",
            "standard": "Tschoibalsan-Normalzeit",
            "daylight": "Tschoibalsan-Sommerzeit"
        }
    },
    "Christmas": {
        "long": {
            "standard": "Weihnachtsinsel-Zeit"
        }
    },
    "Cocos": {
        "long": {
            "standard": "Kokosinseln-Zeit"
        }
    }

```

```

    }
  },
  "Colombia": {
    "long": {
      "generic": "Kolumbianische Zeit",
      "standard": "Kolumbianische Normalzeit",
      "daylight": "Kolumbianische Sommerzeit"
    }
  },
  "Cook": {
    "long": {
      "generic": "Cookinseln-Zeit",
      "standard": "Cookinseln-Normalzeit",
      "daylight": "Cookinseln-Sommerzeit"
    }
  },
  "Cuba": {
    "long": {
      "generic": "Kubanische Zeit",
      "standard": "Kubanische Normalzeit",
      "daylight": "Kubanische Sommerzeit"
    }
  },
  "Davis": {
    "long": {
      "standard": "Davis-Zeit"
    }
  },
  "DumontDUrville": {
    "long": {
      "standard": "Dumont-d'Urville-Zeit"
    }
  },
  "East_Timor": {
    "long": {
      "standard": "Osttimor-Zeit"
    }
  },
  "Easter": {
    "long": {
      "generic": "Osterinsel-Zeit",
      "standard": "Osterinsel-Normalzeit",
      "daylight": "Osterinsel-Sommerzeit"
    }
  },
  "Ecuador": {
    "long": {
      "standard": "Ecuadorianische Zeit"
    }
  },
  "Europe_Central": {
    "long": {
      "generic": "Mitteleuropäische Zeit",
      "standard": "Mitteleuropäische Normalzeit",
      "daylight": "Mitteleuropäische Sommerzeit"
    },
    "short": {

```



```

        "generic": "MEZ",
        "standard": "MEZ",
        "daylight": "MESZ"
    },
},
"Europe_Eastern": {
    "long": {
        "generic": "Osteuropäische Zeit",
        "standard": "Osteuropäische Normalzeit",
        "daylight": "Osteuropäische Sommerzeit"
    },
    "short": {
        "generic": "OEZ",
        "standard": "OEZ",
        "daylight": "OESZ"
    }
},
"Europe_Further_Eastern": {
    "long": {
        "standard": "Kaliningrader Zeit"
    }
},
"Europe_Western": {
    "long": {
        "generic": "Westeuropäische Zeit",
        "standard": "Westeuropäische Normalzeit",
        "daylight": "Westeuropäische Sommerzeit"
    },
    "short": {
        "generic": "WEZ",
        "standard": "WEZ",
        "daylight": "WESZ"
    }
},
"Falkland": {
    "long": {
        "generic": "Falklandinseln-Zeit",
        "standard": "Falklandinseln-Normalzeit",
        "daylight": "Falklandinseln-Sommerzeit"
    }
},
"Fiji": {
    "long": {
        "generic": "Fidschi-Zeit",
        "standard": "Fidschi-Normalzeit",
        "daylight": "Fidschi-Sommerzeit"
    }
},
"French_Guiana": {
    "long": {
        "standard": "Französisch-Guayana-Zeit"
    }
},
"French_Southern": {
    "long": {
        "standard": "Französische Süd- und Antarktisgebiete-Zeit"
    }
}

```

```

    },
    "Galapagos": {
      "long": {
        "standard": "Galapagos-Zeit"
      }
    },
    "Gambier": {
      "long": {
        "standard": "Gambier-Zeit"
      }
    },
    "Georgia": {
      "long": {
        "generic": "Georgische Zeit",
        "standard": "Georgische Normalzeit",
        "daylight": "Georgische Sommerzeit"
      }
    },
    "Gilbert_Islands": {
      "long": {
        "standard": "Gilbert-Inseln-Zeit"
      }
    },
    "GMT": {
      "long": {
        "standard": "Mittlere Greenwich-Zeit"
      }
    },
    "Greenland_Eastern": {
      "long": {
        "generic": "Ostgrönland-Zeit",
        "standard": "Ostgrönland-Normalzeit",
        "daylight": "Ostgrönland-Sommerzeit"
      }
    },
    "Greenland_Western": {
      "long": {
        "generic": "Westgrönland-Zeit",
        "standard": "Westgrönland-Normalzeit",
        "daylight": "Westgrönland-Sommerzeit"
      }
    },
    "Guam": {
      "long": {
        "standard": "Guam-Zeit"
      }
    },
    "Gulf": {
      "long": {
        "standard": "Golf-Zeit"
      }
    },
    "Guyana": {
      "long": {
        "standard": "Guyana-Zeit"
      }
    },
  },

```

```
"Hawaii_Aleutian": {
  "long": {
    "generic": "Hawaii-Aleuten-Zeit",
    "standard": "Hawaii-Aleuten-Normalzeit",
    "daylight": "Hawaii-Aleuten-Sommerzeit"
  }
},
"Hong_Kong": {
  "long": {
    "generic": "Hongkong-Zeit",
    "standard": "Hongkong-Normalzeit",
    "daylight": "Hongkong-Sommerzeit"
  }
},
"Hovd": {
  "long": {
    "generic": "Chowd-Zeit",
    "standard": "Chowd-Normalzeit",
    "daylight": "Chowd-Sommerzeit"
  }
},
"India": {
  "long": {
    "standard": "Indische Zeit"
  }
},
"Indian_Ocean": {
  "long": {
    "standard": "Indischer Ozean-Zeit"
  }
},
"Indochina": {
  "long": {
    "standard": "Indochina-Zeit"
  }
},
"Indonesia_Central": {
  "long": {
    "standard": "Zentralindonesische Zeit"
  }
},
"Indonesia_Eastern": {
  "long": {
    "standard": "Ostindonesische Zeit"
  }
},
"Indonesia_Western": {
  "long": {
    "standard": "Westindonesische Zeit"
  }
},
"Iran": {
  "long": {
    "generic": "Iranische Zeit",
    "standard": "Iranische Normalzeit",
    "daylight": "Iranische Sommerzeit"
  }
}
```

```
    },
    "Irkutsk": {
      "long": {
        "generic": "Irkutsk-Zeit",
        "standard": "Irkutsk-Normalzeit",
        "daylight": "Irkutsk-Sommerzeit"
      }
    },
    "Israel": {
      "long": {
        "generic": "Israelische Zeit",
        "standard": "Israelische Normalzeit",
        "daylight": "Israelische Sommerzeit"
      }
    },
    "Japan": {
      "long": {
        "generic": "Japanische Zeit",
        "standard": "Japanische Normalzeit",
        "daylight": "Japanische Sommerzeit"
      }
    },
    "Kamchatka": {
      "long": {
        "generic": "Kamtschatka-Zeit",
        "standard": "Kamtschatka-Normalzeit",
        "daylight": "Kamtschatka-Sommerzeit"
      }
    },
    "Kazakhstan_Eastern": {
      "long": {
        "standard": "Ostkasachische Zeit"
      }
    },
    "Kazakhstan_Western": {
      "long": {
        "standard": "Westkasachische Zeit"
      }
    },
    "Korea": {
      "long": {
        "generic": "Koreanische Zeit",
        "standard": "Koreanische Normalzeit",
        "daylight": "Koreanische Sommerzeit"
      }
    },
    "Kosrae": {
      "long": {
        "standard": "Kosrae-Zeit"
      }
    },
    "Krasnoyarsk": {
      "long": {
        "generic": "Krasnojarsk-Zeit",
        "standard": "Krasnojarsk-Normalzeit",
        "daylight": "Krasnojarsk-Sommerzeit"
      }
    }
  }
```

```
    },
    "Kyrgystan": {
      "long": {
        "standard": "Kirgisistan-Zeit"
      }
    },
    "Lanka": {
      "long": {
        "standard": "Sri-Lanka-Zeit"
      }
    },
    "Line_Islands": {
      "long": {
        "standard": "Linieninseln-Zeit"
      }
    },
    "Lord_Howe": {
      "long": {
        "generic": "Lord-Howe-Zeit",
        "standard": "Lord-Howe-Normalzeit",
        "daylight": "Lord-Howe-Sommerzeit"
      }
    },
    "Macau": {
      "long": {
        "generic": "Macau-Zeit",
        "standard": "Macau-Normalzeit",
        "daylight": "Macau-Sommerzeit"
      }
    },
    "Macquarie": {
      "long": {
        "standard": "Macquarieinsel-Zeit"
      }
    },
    "Magadan": {
      "long": {
        "generic": "Magadan-Zeit",
        "standard": "Magadan-Normalzeit",
        "daylight": "Magadan-Sommerzeit"
      }
    },
    "Malaysia": {
      "long": {
        "standard": "Malaysische Zeit"
      }
    },
    "Maldives": {
      "long": {
        "standard": "Malediven-Zeit"
      }
    },
    "Marquesas": {
      "long": {
        "standard": "Marquesas-Zeit"
      }
    }
  },
}
```

```
"Marshall_Islands": {
  "long": {
    "standard": "Marshallinseln-Zeit"
  }
},
"Mauritius": {
  "long": {
    "generic": "Mauritius-Zeit",
    "standard": "Mauritius-Normalzeit",
    "daylight": "Mauritius-Sommerzeit"
  }
},
"Mawson": {
  "long": {
    "standard": "Mawson-Zeit"
  }
},
"Mexico_Northwest": {
  "long": {
    "generic": "Mexiko Nordwestliche Zone-Zeit",
    "standard": "Mexiko Nordwestliche Zone-Normalzeit",
    "daylight": "Mexiko Nordwestliche Zone-Sommerzeit"
  }
},
"Mexico_Pacific": {
  "long": {
    "generic": "Mexiko Pazifikzone-Zeit",
    "standard": "Mexiko Pazifikzone-Normalzeit",
    "daylight": "Mexiko Pazifikzone-Sommerzeit"
  }
},
"Mongolia": {
  "long": {
    "generic": "Ulaanbaatar-Zeit",
    "standard": "Ulaanbaatar-Normalzeit",
    "daylight": "Ulaanbaatar-Sommerzeit"
  }
},
"Moscow": {
  "long": {
    "generic": "Moskauer Zeit",
    "standard": "Moskauer Normalzeit",
    "daylight": "Moskauer Sommerzeit"
  }
},
"Myanmar": {
  "long": {
    "standard": "Myanmar-Zeit"
  }
},
"Nauru": {
  "long": {
    "standard": "Nauru-Zeit"
  }
},
"Nepal": {
  "long": {
```

```

        "standard": "Nepalesische Zeit"
    },
    },
    "New_Caledonia": {
        "long": {
            "generic": "Neukaledonische Zeit",
            "standard": "Neukaledonische Normalzeit",
            "daylight": "Neukaledonische Sommerzeit"
        }
    },
    "New_Zealand": {
        "long": {
            "generic": "Neuseeland-Zeit",
            "standard": "Neuseeland-Normalzeit",
            "daylight": "Neuseeland-Sommerzeit"
        }
    },
    "Newfoundland": {
        "long": {
            "generic": "Neufundland-Zeit",
            "standard": "Neufundland-Normalzeit",
            "daylight": "Neufundland-Sommerzeit"
        }
    },
    "Niue": {
        "long": {
            "standard": "Niue-Zeit"
        }
    },
    "Norfolk": {
        "long": {
            "standard": "Norfolkinsel-Zeit"
        }
    },
    "Noronha": {
        "long": {
            "generic": "Fernando de Noronha-Zeit",
            "standard": "Fernando de Noronha-Normalzeit",
            "daylight": "Fernando de Noronha-Sommerzeit"
        }
    },
    "North_Mariana": {
        "long": {
            "standard": "Nördliche-Marianen-Zeit"
        }
    },
    "Novosibirsk": {
        "long": {
            "generic": "Nowosibirsk-Zeit",
            "standard": "Nowosibirsk-Normalzeit",
            "daylight": "Nowosibirsk-Sommerzeit"
        }
    },
    "Omsk": {
        "long": {
            "generic": "Omsk-Zeit",
            "standard": "Omsk-Normalzeit",

```

```

        "daylight": "Omsk-Sommerzeit"
    },
    },
    "Pakistan": {
        "long": {
            "generic": "Pakistanische Zeit",
            "standard": "Pakistanische Normalzeit",
            "daylight": "Pakistanische Sommerzeit"
        }
    },
    "Palau": {
        "long": {
            "standard": "Palau-Zeit"
        }
    },
    },
    "Papua_New_Guinea": {
        "long": {
            "standard": "Papua-Neuguinea-Zeit"
        }
    },
    },
    "Paraguay": {
        "long": {
            "generic": "Paraguayische Zeit",
            "standard": "Paraguayische Normalzeit",
            "daylight": "Paraguayische Sommerzeit"
        }
    },
    },
    "Peru": {
        "long": {
            "generic": "Peruanische Zeit",
            "standard": "Peruanische Normalzeit",
            "daylight": "Peruanische Sommerzeit"
        }
    },
    },
    "Philippines": {
        "long": {
            "generic": "Philippinische Zeit",
            "standard": "Philippinische Normalzeit",
            "daylight": "Philippinische Sommerzeit"
        }
    },
    },
    "Phoenix_Islands": {
        "long": {
            "standard": "Phoenixinseln-Zeit"
        }
    },
    },
    "Pierre_Miquelon": {
        "long": {
            "generic": "Saint-Pierre-und-Miquelon-Zeit",
            "standard": "Saint-Pierre-und-Miquelon-Normalzeit",
            "daylight": "Saint-Pierre-und-Miquelon-Sommerzeit"
        }
    },
    },
    "Pitcairn": {
        "long": {
            "standard": "Pitcairnsinseln-Zeit"
        }
    }
}

```



```
    },
    "Ponape": {
      "long": {
        "standard": "Ponape-Zeit"
      }
    },
    "Pyongyang": {
      "long": {
        "standard": "Pjöngjang-Zeit"
      }
    },
    "Qyzylorda": {
      "long": {
        "generic": "Quysylorda-Zeit",
        "standard": "Quysylorda-Normalzeit",
        "daylight": "Qysylorda-Sommerzeit"
      }
    },
    "Reunion": {
      "long": {
        "standard": "Réunion-Zeit"
      }
    },
    "Rothera": {
      "long": {
        "standard": "Rothera-Zeit"
      }
    },
    "Sakhalin": {
      "long": {
        "generic": "Sachalin-Zeit",
        "standard": "Sachalin-Normalzeit",
        "daylight": "Sachalin-Sommerzeit"
      }
    },
    "Samara": {
      "long": {
        "generic": "Samara-Zeit",
        "standard": "Samara-Normalzeit",
        "daylight": "Samara-Sommerzeit"
      }
    },
    "Samoa": {
      "long": {
        "generic": "Samoa-Zeit",
        "standard": "Samoa-Normalzeit",
        "daylight": "Samoa-Sommerzeit"
      }
    },
    "Seychelles": {
      "long": {
        "standard": "Seychellen-Zeit"
      }
    },
    "Singapore": {
      "long": {
        "standard": "Singapur-Zeit"
      }
    }
  }
```

```
    }  
  },  
  "Solomon": {  
    "long": {  
      "standard": "Salomoninseln-Zeit"  
    }  
  },  
  "South_Georgia": {  
    "long": {  
      "standard": "Südgeorgische Zeit"  
    }  
  },  
  "Suriname": {  
    "long": {  
      "standard": "Suriname-Zeit"  
    }  
  },  
  "Syowa": {  
    "long": {  
      "standard": "Syowa-Zeit"  
    }  
  },  
  "Tahiti": {  
    "long": {  
      "standard": "Tahiti-Zeit"  
    }  
  },  
  "Taipei": {  
    "long": {  
      "generic": "Taipeh-Zeit",  
      "standard": "Taipeh-Normalzeit",  
      "daylight": "Taipeh-Sommerzeit"  
    }  
  },  
  "Tajikistan": {  
    "long": {  
      "standard": "Tadschikistan-Zeit"  
    }  
  },  
  "Tokelau": {  
    "long": {  
      "standard": "Tokelau-Zeit"  
    }  
  },  
  "Tonga": {  
    "long": {  
      "generic": "Tonganische Zeit",  
      "standard": "Tonganische Normalzeit",  
      "daylight": "Tonganische Sommerzeit"  
    }  
  },  
  "Truk": {  
    "long": {  
      "standard": "Chuuk-Zeit"  
    }  
  },  
  "Turkmenistan": {
```

```

        "long": {
            "generic": "Turkmenistan-Zeit",
            "standard": "Turkmenistan-Normalzeit",
            "daylight": "Turkmenistan-Sommerzeit"
        }
    },
    "Tuvalu": {
        "long": {
            "standard": "Tuvalu-Zeit"
        }
    },
    "Uruguay": {
        "long": {
            "generic": "Uruguayische Zeit",
            "standard": "Uruguayische Normalzeit",
            "daylight": "Uruguayische Sommerzeit"
        }
    },
    "Uzbekistan": {
        "long": {
            "generic": "Usbekistan-Zeit",
            "standard": "Usbekistan-Normalzeit",
            "daylight": "Usbekistan-Sommerzeit"
        }
    },
    "Vanuatu": {
        "long": {
            "generic": "Vanuatu-Zeit",
            "standard": "Vanuatu-Normalzeit",
            "daylight": "Vanuatu-Sommerzeit"
        }
    },
    "Venezuela": {
        "long": {
            "standard": "Venezuela-Zeit"
        }
    },
    "Vladivostok": {
        "long": {
            "generic": "Wladiwostok-Zeit",
            "standard": "Wladiwostok-Normalzeit",
            "daylight": "Wladiwostok-Sommerzeit"
        }
    },
    "Volgograd": {
        "long": {
            "generic": "Wolgograd-Zeit",
            "standard": "Wolgograd-Normalzeit",
            "daylight": "Wolgograd-Sommerzeit"
        }
    },
    "Vostok": {
        "long": {
            "standard": "Wostok-Zeit"
        }
    },
    "Wake": {

```

```

        "long": {
          "standard": "Wake-Insel-Zeit"
        }
      },
      "Wallis": {
        "long": {
          "standard": "Wallis-und-Futuna-Zeit"
        }
      },
      "Yakutsk": {
        "long": {
          "generic": "Jakutsk-Zeit",
          "standard": "Jakutsk-Normalzeit",
          "daylight": "Jakutsk-Sommerzeit"
        }
      },
      "Yekaterinburg": {
        "long": {
          "generic": "Jekaterinburg-Zeit",
          "standard": "Jekaterinburg-Normalzeit",
          "daylight": "Jekaterinburg-Sommerzeit"
        }
      }
    }
  }
}

```

TIMEZONENAMES.JSX

```

{
  "main";
  {
    "de";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 12879 $",
          "_cldrVersion";
          "30.0.3";
        }
        "language";
        "de";
      }
      "dates";
      {
        "timeZoneNames";
        {
          "hourFormat";
          "+HH:mm;-HH:mm",
          "gmtFormat";

```

```

"GMT{0}",
  "gmtZeroFormat";
"GMT",
  "regionFormat";
"{0} Zeit",
  "regionFormat-type-daylight";
"{0} Sommerzeit",
  "regionFormat-type-standard";
"{0} Normalzeit",
  "fallbackFormat";
"{1} ({0})",
  "zone";
{
  "America";
  {
    "Adak";
    {
      "exemplarCity";
      "Adak";
    }
    "Anchorage";
    {
      "exemplarCity";
      "Anchorage";
    }
    "Anguilla";
    {
      "exemplarCity";
      "Anguilla";
    }
    "Antigua";
    {
      "exemplarCity";
      "Antigua";
    }
    "Araguaina";
    {
      "exemplarCity";
      "Araguaina";
    }
    "Argentina";
    {
      "Rio_Gallegos";
      {
        "exemplarCity";
        "Rio Gallegos";
      }
      "San_Juan";
      {
        "exemplarCity";
        "San Juan";
      }
      "Ushuaia";
      {
        "exemplarCity";
        "Ushuaia";
      }
    }
  }
}

```

```
"La_Rioja";
{
  "exemplarCity";
  "La Rioja";
}
"San_Luis";
{
  "exemplarCity";
  "San Luis";
}
"Salta";
{
  "exemplarCity";
  "Salta";
}
"Tucuman";
{
  "exemplarCity";
  "Tucuman";
}
}
"Aruba";
{
  "exemplarCity";
  "Aruba";
}
"Asuncion";
{
  "exemplarCity";
  "Asunción";
}
"Bahia";
{
  "exemplarCity";
  "Bahia";
}
"Bahia_Banderas";
{
  "exemplarCity";
  "Bahia Banderas";
}
"Barbados";
{
  "exemplarCity";
  "Barbados";
}
"Belem";
{
  "exemplarCity";
  "Belem";
}
"Belize";
{
  "exemplarCity";
  "Belize";
}
}
"Blanc-Sablon";
```

```
{
    "exemplarCity";
    "Blanc-Sablon";
}
"Boa_Vista";
{
    "exemplarCity";
    "Boa Vista";
}
"Bogota";
{
    "exemplarCity";
    "Bogotá";
}
"Boise";
{
    "exemplarCity";
    "Boise";
}
"Buenos_Aires";
{
    "exemplarCity";
    "Buenos Aires";
}
"Cambridge_Bay";
{
    "exemplarCity";
    "Cambridge Bay";
}
"Campo_Grande";
{
    "exemplarCity";
    "Campo Grande";
}
"Cancun";
{
    "exemplarCity";
    "Cancún";
}
"Caracas";
{
    "exemplarCity";
    "Caracas";
}
"Catamarca";
{
    "exemplarCity";
    "Catamarca";
}
"Cayenne";
{
    "exemplarCity";
    "Cayenne";
}
"Cayman";
{
    "exemplarCity";
```

```
        "Kaimaninseln";
    }
    "Chicago";
    {
        "exemplarCity";
        "Chicago";
    }
    "Chihuahua";
    {
        "exemplarCity";
        "Chihuahua";
    }
    "Coral_Harbour";
    {
        "exemplarCity";
        "Atikokan";
    }
    "Cordoba";
    {
        "exemplarCity";
        "Córdoba";
    }
    "Costa_Rica";
    {
        "exemplarCity";
        "Costa Rica";
    }
    "Creston";
    {
        "exemplarCity";
        "Creston";
    }
    "Cuiaba";
    {
        "exemplarCity";
        "Cuiaba";
    }
    "Curacao";
    {
        "exemplarCity";
        "Curaçao";
    }
    "Danmarkshavn";
    {
        "exemplarCity";
        "Danmarkshavn";
    }
    "Dawson";
    {
        "exemplarCity";
        "Dawson";
    }
    "Dawson_Creek";
    {
        "exemplarCity";
        "Dawson Creek";
    }
}
```



```
"Denver";
{
  "exemplarCity";
  "Denver";
}
"Detroit";
{
  "exemplarCity";
  "Detroit";
}
"Dominica";
{
  "exemplarCity";
  "Dominica";
}
"Edmonton";
{
  "exemplarCity";
  "Edmonton";
}
"Eirunepe";
{
  "exemplarCity";
  "Eirunepe";
}
"El_Salvador";
{
  "exemplarCity";
  "El Salvador";
}
"Fort_Nelson";
{
  "exemplarCity";
  "Fort Nelson";
}
"Fortaleza";
{
  "exemplarCity";
  "Fortaleza";
}
"Glace_Bay";
{
  "exemplarCity";
  "Glace Bay";
}
"Godthab";
{
  "exemplarCity";
  "Nuuk";
}
"Goose_Bay";
{
  "exemplarCity";
  "Goose Bay";
}
"Grand_Turk";
{
```

```
        "exemplarCity";
        "Grand Turk";
    }
    "Grenada";
    {
        "exemplarCity";
        "Grenada";
    }
    "Guadeloupe";
    {
        "exemplarCity";
        "Guadeloupe";
    }
    "Guatemala";
    {
        "exemplarCity";
        "Guatemala";
    }
    "Guayaquil";
    {
        "exemplarCity";
        "Guayaquil";
    }
    "Guyana";
    {
        "exemplarCity";
        "Guyana";
    }
    "Halifax";
    {
        "exemplarCity";
        "Halifax";
    }
    "Havana";
    {
        "exemplarCity";
        "Havanna";
    }
    "Hermosillo";
    {
        "exemplarCity";
        "Hermosillo";
    }
    "Indiana";
    {
        "Vincennes";
        {
            "exemplarCity";
            "Vincennes, Indiana";
        }
        "Petersburg";
        {
            "exemplarCity";
            "Petersburg, Indiana";
        }
        "Tell_City";
        {
```

```
        "exemplarCity";
        "Tell City, Indiana";
    }
    "Knox";
    {
        "exemplarCity";
        "Knox, Indiana";
    }
    "Winamac";
    {
        "exemplarCity";
        "Winamac, Indiana";
    }
    "Marengo";
    {
        "exemplarCity";
        "Marengo, Indiana";
    }
    "Vevay";
    {
        "exemplarCity";
        "Vevay, Indiana";
    }
}
"Indianapolis";
{
    "exemplarCity";
    "Indianapolis";
}
"Inuvik";
{
    "exemplarCity";
    "Inuvik";
}
"Iqaluit";
{
    "exemplarCity";
    "Iqaluit";
}
"Jamaica";
{
    "exemplarCity";
    "Jamaika";
}
"Jujuy";
{
    "exemplarCity";
    "Jujuy";
}
"Juneau";
{
    "exemplarCity";
    "Juneau";
}
"Kentucky";
{
    "Monticello";
```

```
        {
            "exemplarCity";
            "Monticello, Kentucky";
        }
    }
    "Kralendijk";
    {
        "exemplarCity";
        "Kralendijk";
    }
    "La_Paz";
    {
        "exemplarCity";
        "La Paz";
    }
    "Lima";
    {
        "exemplarCity";
        "Lima";
    }
    "Los_Angeles";
    {
        "exemplarCity";
        "Los Angeles";
    }
    "Louisville";
    {
        "exemplarCity";
        "Louisville";
    }
    "Lower_Princes";
    {
        "exemplarCity";
        "Lower Prince's Quarter";
    }
    "Maceio";
    {
        "exemplarCity";
        "Maceio";
    }
    "Managua";
    {
        "exemplarCity";
        "Managua";
    }
    "Manaus";
    {
        "exemplarCity";
        "Manaus";
    }
    "Marigot";
    {
        "exemplarCity";
        "Marigot";
    }
    "Martinique";
    {
```

```
        "exemplarCity";
        "Martinique";
    }
    "Matamoros";
    {
        "exemplarCity";
        "Matamoros";
    }
    "Mazatlan";
    {
        "exemplarCity";
        "Mazatlan";
    }
    "Mendoza";
    {
        "exemplarCity";
        "Mendoza";
    }
    "Menominee";
    {
        "exemplarCity";
        "Menominee";
    }
    "Merida";
    {
        "exemplarCity";
        "Merida";
    }
    "Metlakatla";
    {
        "exemplarCity";
        "Metlakatla";
    }
    "Mexico_City";
    {
        "exemplarCity";
        "Mexiko-Stadt";
    }
    "Miquelon";
    {
        "exemplarCity";
        "Miquelon";
    }
    "Moncton";
    {
        "exemplarCity";
        "Moncton";
    }
    "Monterrey";
    {
        "exemplarCity";
        "Monterrey";
    }
    "Montevideo";
    {
        "exemplarCity";
        "Montevideo";
    }
```

```
}
"Montserrat";
{
  "exemplarCity";
  "Montserrat";
}
"Nassau";
{
  "exemplarCity";
  "Nassau";
}
"New_York";
{
  "exemplarCity";
  "New York";
}
"Nipigon";
{
  "exemplarCity";
  "Nipigon";
}
"Nome";
{
  "exemplarCity";
  "Nome";
}
"Noronha";
{
  "exemplarCity";
  "Noronha";
}
"North_Dakota";
{
  "Beulah";
  {
    "exemplarCity";
    "Beulah, North Dakota";
  }
  "New_Salem";
  {
    "exemplarCity";
    "New Salem, North Dakota";
  }
  "Center";
  {
    "exemplarCity";
    "Center, North Dakota";
  }
}
"Ojinaga";
{
  "exemplarCity";
  "Ojinaga";
}
"Panama";
{
  "exemplarCity";
```

```
        "Panama";
    }
    "Pangnirtung";
    {
        "exemplarCity";
        "Pangnirtung";
    }
    "Paramaribo";
    {
        "exemplarCity";
        "Paramaribo";
    }
    "Phoenix";
    {
        "exemplarCity";
        "Phoenix";
    }
    "Port-au-Prince";
    {
        "exemplarCity";
        "Port-au-Prince";
    }
    "Port_of_Spain";
    {
        "exemplarCity";
        "Port of Spain";
    }
    "Porto_Velho";
    {
        "exemplarCity";
        "Porto Velho";
    }
    "Puerto_Rico";
    {
        "exemplarCity";
        "Puerto Rico";
    }
    "Rainy_River";
    {
        "exemplarCity";
        "Rainy River";
    }
    "Rankin_Inlet";
    {
        "exemplarCity";
        "Rankin Inlet";
    }
    "Recife";
    {
        "exemplarCity";
        "Recife";
    }
    "Regina";
    {
        "exemplarCity";
        "Regina";
    }
}
```

```
"Resolute";
{
  "exemplarCity";
  "Resolute";
}
"Rio_Branco";
{
  "exemplarCity";
  "Rio Branco";
}
"Santa_Isabel";
{
  "exemplarCity";
  "Santa Isabel";
}
"Santarem";
{
  "exemplarCity";
  "Santarem";
}
"Santiago";
{
  "exemplarCity";
  "Santiago";
}
"Santo_Domingo";
{
  "exemplarCity";
  "Santo Domingo";
}
"Sao_Paulo";
{
  "exemplarCity";
  "São Paulo";
}
"Scoresbysund";
{
  "exemplarCity";
  "Ittoqqortoormiit";
}
"Sitka";
{
  "exemplarCity";
  "Sitka";
}
"St_Barthelemy";
{
  "exemplarCity";
  "Saint-Barthélemy";
}
"St_Johns";
{
  "exemplarCity";
  "St. John's";
}
"St_Kitts";
{
```



```
        "exemplarCity";
        "St. Kitts";
    }
    "St_Lucia";
    {
        "exemplarCity";
        "St. Lucia";
    }
    "St_Thomas";
    {
        "exemplarCity";
        "St. Thomas";
    }
    "St_Vincent";
    {
        "exemplarCity";
        "St. Vincent";
    }
    "Swift_Current";
    {
        "exemplarCity";
        "Swift Current";
    }
    "Tegucigalpa";
    {
        "exemplarCity";
        "Tegucigalpa";
    }
    "Thule";
    {
        "exemplarCity";
        "Thule";
    }
    "Thunder_Bay";
    {
        "exemplarCity";
        "Thunder Bay";
    }
    "Tijuana";
    {
        "exemplarCity";
        "Tijuana";
    }
    "Toronto";
    {
        "exemplarCity";
        "Toronto";
    }
    "Tortola";
    {
        "exemplarCity";
        "Tortola";
    }
    "Vancouver";
    {
        "exemplarCity";
        "Vancouver";
    }
```

```
}
  "Whitehorse";
  {
    "exemplarCity";
    "Whitehorse";
  }
  "Winnipeg";
  {
    "exemplarCity";
    "Winnipeg";
  }
  "Yakutat";
  {
    "exemplarCity";
    "Yakutat";
  }
  "Yellowknife";
  {
    "exemplarCity";
    "Yellowknife";
  }
}
"Atlantic";
{
  "Azores";
  {
    "exemplarCity";
    "Azoren";
  }
  "Bermuda";
  {
    "exemplarCity";
    "Bermudas";
  }
  "Canary";
  {
    "exemplarCity";
    "Kanaren";
  }
  "Cape_Verde";
  {
    "exemplarCity";
    "Cabo Verde";
  }
  "Faeroe";
  {
    "exemplarCity";
    "Färöer";
  }
  "Madeira";
  {
    "exemplarCity";
    "Madeira";
  }
  "Reykjavik";
  {
    "exemplarCity";
```

```
        "Reykjavík";
    }
    "South_Georgia";
    {
        "exemplarCity";
        "Südgeorgien";
    }
    "St_Helena";
    {
        "exemplarCity";
        "St. Helena";
    }
    "Stanley";
    {
        "exemplarCity";
        "Stanley";
    }
}
"Europe";
{
    "Amsterdam";
    {
        "exemplarCity";
        "Amsterdam";
    }
    "Andorra";
    {
        "exemplarCity";
        "Andorra";
    }
    "Astrakhan";
    {
        "exemplarCity";
        "Astrachan";
    }
    "Athens";
    {
        "exemplarCity";
        "Athen";
    }
    "Belgrade";
    {
        "exemplarCity";
        "Belgrad";
    }
    "Berlin";
    {
        "exemplarCity";
        "Berlin";
    }
    "Bratislava";
    {
        "exemplarCity";
        "Bratislava";
    }
    "Brussels";
    {
```

```
        "exemplarCity";
        "Brüssel";
    }
    "Bucharest";
    {
        "exemplarCity";
        "Bukarest";
    }
    "Budapest";
    {
        "exemplarCity";
        "Budapest";
    }
    "Busingen";
    {
        "exemplarCity";
        "Büsingen";
    }
    "Chisinau";
    {
        "exemplarCity";
        "Kischinau";
    }
    "Copenhagen";
    {
        "exemplarCity";
        "Kopenhagen";
    }
    "Dublin";
    {
        "long";
        {
            "daylight";
            "Irische Sommerzeit";
        }
        "exemplarCity";
        "Dublin";
    }
    "Gibraltar";
    {
        "exemplarCity";
        "Gibraltar";
    }
    "Guernsey";
    {
        "exemplarCity";
        "Guernsey";
    }
    "Helsinki";
    {
        "exemplarCity";
        "Helsinki";
    }
    "Isle_of_Man";
    {
        "exemplarCity";
        "Isle of Man";
    }
}
```

```
}
"Istanbul";
{
  "exemplarCity";
  "Istanbul";
}
"Jersey";
{
  "exemplarCity";
  "Jersey";
}
"Kaliningrad";
{
  "exemplarCity";
  "Kaliningrad";
}
"Kiev";
{
  "exemplarCity";
  "Kiew";
}
"Kirov";
{
  "exemplarCity";
  "Kirow";
}
"Lisbon";
{
  "exemplarCity";
  "Lissabon";
}
"Ljubljana";
{
  "exemplarCity";
  "Ljubljana";
}
"London";
{
  "long";
  {
    "daylight";
    "Britische Sommerzeit";
  }
  "exemplarCity";
  "London";
}
"Luxembourg";
{
  "exemplarCity";
  "Luxemburg";
}
"Madrid";
{
  "exemplarCity";
  "Madrid";
}
"Malta";
```

```
{
  "exemplarCity";
  "Malta";
}
"Mariehamn";
{
  "exemplarCity";
  "Mariehamn";
}
"Minsk";
{
  "exemplarCity";
  "Minsk";
}
"Monaco";
{
  "exemplarCity";
  "Monaco";
}
"Moscow";
{
  "exemplarCity";
  "Moskau";
}
"Oslo";
{
  "exemplarCity";
  "Oslo";
}
"Paris";
{
  "exemplarCity";
  "Paris";
}
"Podgorica";
{
  "exemplarCity";
  "Podgorica";
}
"Prague";
{
  "exemplarCity";
  "Prag";
}
"Riga";
{
  "exemplarCity";
  "Riga";
}
"Rome";
{
  "exemplarCity";
  "Rom";
}
"Samara";
{
  "exemplarCity";
```

```
        "Samara";
    }
    "San_Marino";
    {
        "exemplarCity";
        "San Marino";
    }
    "Sarajevo";
    {
        "exemplarCity";
        "Sarajevo";
    }
    "Simferopol";
    {
        "exemplarCity";
        "Simferopol";
    }
    "Skopje";
    {
        "exemplarCity";
        "Skopje";
    }
    "Sofia";
    {
        "exemplarCity";
        "Sofia";
    }
    "Stockholm";
    {
        "exemplarCity";
        "Stockholm";
    }
    "Tallinn";
    {
        "exemplarCity";
        "Tallinn";
    }
    "Tirane";
    {
        "exemplarCity";
        "Tirana";
    }
    "Ulyanovsk";
    {
        "exemplarCity";
        "Uljanowsk";
    }
    "Uzhgorod";
    {
        "exemplarCity";
        "Uschgorod";
    }
    "Vaduz";
    {
        "exemplarCity";
        "Vaduz";
    }
}
```

```
"Vatican";
{
  "exemplarCity";
  "Vatikan";
}
"Vienna";
{
  "exemplarCity";
  "Wien";
}
"Vilnius";
{
  "exemplarCity";
  "Vilnius";
}
"Volgograd";
{
  "exemplarCity";
  "Wolgograd";
}
"Warsaw";
{
  "exemplarCity";
  "Warschau";
}
"Zagreb";
{
  "exemplarCity";
  "Zagreb";
}
"Zaporozhye";
{
  "exemplarCity";
  "Saporischja";
}
"Zurich";
{
  "exemplarCity";
  "Zürich";
}
}
"Africa";
{
  "Abidjan";
  {
    "exemplarCity";
    "Abidjan";
  }
  "Accra";
  {
    "exemplarCity";
    "Accra";
  }
  "Addis_Ababa";
  {
    "exemplarCity";
    "Addis Abeba";
  }
}
```



```
}
"Algiers";
{
  "exemplarCity";
  "Algier";
}
"Asmera";
{
  "exemplarCity";
  "Asmara";
}
"Bamako";
{
  "exemplarCity";
  "Bamako";
}
"Bangui";
{
  "exemplarCity";
  "Bangui";
}
"Banjul";
{
  "exemplarCity";
  "Banjul";
}
"Bissau";
{
  "exemplarCity";
  "Bissau";
}
"Blantyre";
{
  "exemplarCity";
  "Blantyre";
}
"Brazzaville";
{
  "exemplarCity";
  "Brazzaville";
}
"Bujumbura";
{
  "exemplarCity";
  "Bujumbura";
}
"Cairo";
{
  "exemplarCity";
  "Kairo";
}
"Casablanca";
{
  "exemplarCity";
  "Casablanca";
}
"Ceuta";
```

```
{
    "exemplarCity";
    "Ceuta";
}
"Conakry";
{
    "exemplarCity";
    "Conakry";
}
"Dakar";
{
    "exemplarCity";
    "Dakar";
}
"Dar_es_Salaam";
{
    "exemplarCity";
    "Daressalam";
}
"Djibouti";
{
    "exemplarCity";
    "Dschibuti";
}
"Douala";
{
    "exemplarCity";
    "Douala";
}
"El_Aaiun";
{
    "exemplarCity";
    "El Aaiún";
}
"Freetown";
{
    "exemplarCity";
    "Freetown";
}
"Gaborone";
{
    "exemplarCity";
    "Gaborone";
}
"Harare";
{
    "exemplarCity";
    "Harare";
}
"Johannesburg";
{
    "exemplarCity";
    "Johannesburg";
}
"Juba";
{
    "exemplarCity";
```

```
        "Juba";
    }
    "Kampala";
    {
        "exemplarCity";
        "Kampala";
    }
    "Khartoum";
    {
        "exemplarCity";
        "Khartum";
    }
    "Kigali";
    {
        "exemplarCity";
        "Kigali";
    }
    "Kinshasa";
    {
        "exemplarCity";
        "Kinshasa";
    }
    "Lagos";
    {
        "exemplarCity";
        "Lagos";
    }
    "Libreville";
    {
        "exemplarCity";
        "Libreville";
    }
    "Lome";
    {
        "exemplarCity";
        "Lomé";
    }
    "Luanda";
    {
        "exemplarCity";
        "Luanda";
    }
    "Lubumbashi";
    {
        "exemplarCity";
        "Lubumbashi";
    }
    "Lusaka";
    {
        "exemplarCity";
        "Lusaka";
    }
    "Malabo";
    {
        "exemplarCity";
        "Malabo";
    }
}
```

```
"Maputo";
{
  "exemplarCity";
  "Maputo";
}
"Maseru";
{
  "exemplarCity";
  "Maseru";
}
"Mbabane";
{
  "exemplarCity";
  "Mbabane";
}
"Mogadishu";
{
  "exemplarCity";
  "Mogadishu";
}
"Monrovia";
{
  "exemplarCity";
  "Monrovia";
}
"Nairobi";
{
  "exemplarCity";
  "Nairobi";
}
"Ndjamena";
{
  "exemplarCity";
  "N'Djamena";
}
"Niamey";
{
  "exemplarCity";
  "Niamey";
}
"Nouakchott";
{
  "exemplarCity";
  "Nouakchott";
}
"Ouagadougou";
{
  "exemplarCity";
  "Ouagadougou";
}
"Porto-Novo";
{
  "exemplarCity";
  "Porto Novo";
}
"Sao_Tome";
{
```

```
        "exemplarCity";
        "São Tomé";
    }
    "Tripoli";
    {
        "exemplarCity";
        "Tripolis";
    }
    "Tunis";
    {
        "exemplarCity";
        "Tunis";
    }
    "Windhoek";
    {
        "exemplarCity";
        "Windhoek";
    }
}
"Asia";
{
    "Aden";
    {
        "exemplarCity";
        "Aden";
    }
    "Almaty";
    {
        "exemplarCity";
        "Almaty";
    }
    "Amman";
    {
        "exemplarCity";
        "Amman";
    }
    "Anadyr";
    {
        "exemplarCity";
        "Anadyr";
    }
    "Aqtau";
    {
        "exemplarCity";
        "Aqtau";
    }
    "Aqtobe";
    {
        "exemplarCity";
        "Aktobe";
    }
    "Ashgabat";
    {
        "exemplarCity";
        "Aşgabat";
    }
    "Baghdad";
```

```
{
    "exemplarCity";
    "Bagdad";
}
"Bahrain";
{
    "exemplarCity";
    "Bahrain";
}
"Baku";
{
    "exemplarCity";
    "Baku";
}
"Bangkok";
{
    "exemplarCity";
    "Bangkok";
}
"Barnaul";
{
    "exemplarCity";
    "Barnaul";
}
"Beirut";
{
    "exemplarCity";
    "Beirut";
}
"Bishkek";
{
    "exemplarCity";
    "Bischkek";
}
"Brunei";
{
    "exemplarCity";
    "Brunei";
}
"Calcutta";
{
    "exemplarCity";
    "Kalkutta";
}
"Chita";
{
    "exemplarCity";
    "Tschita";
}
"Choibalsan";
{
    "exemplarCity";
    "Tschoibalsan";
}
"Colombo";
{
    "exemplarCity";
```

```
        "Colombo";
    }
    "Damascus";
    {
        "exemplarCity";
        "Damaskus";
    }
    "Dhaka";
    {
        "exemplarCity";
        "Dhaka";
    }
    "Dili";
    {
        "exemplarCity";
        "Dili";
    }
    "Dubai";
    {
        "exemplarCity";
        "Dubai";
    }
    "Dushanbe";
    {
        "exemplarCity";
        "Duschanbe";
    }
    "Gaza";
    {
        "exemplarCity";
        "Gaza";
    }
    "Hebron";
    {
        "exemplarCity";
        "Hebron";
    }
    "Hong_Kong";
    {
        "exemplarCity";
        "Hongkong";
    }
    "Hovd";
    {
        "exemplarCity";
        "Chowd";
    }
    "Irkutsk";
    {
        "exemplarCity";
        "Irkutsk";
    }
    "Jakarta";
    {
        "exemplarCity";
        "Jakarta";
    }
}
```

```
"Jayapura";
{
  "exemplarCity";
  "Jayapura";
}
"Jerusalem";
{
  "exemplarCity";
  "Jerusalem";
}
"Kabul";
{
  "exemplarCity";
  "Kabul";
}
"Kamchatka";
{
  "exemplarCity";
  "Kamtschatka";
}
"Karachi";
{
  "exemplarCity";
  "Karatschi";
}
"Katmandu";
{
  "exemplarCity";
  "Kathmandu";
}
"Khandyga";
{
  "exemplarCity";
  "Chandyga";
}
"Krasnoyarsk";
{
  "exemplarCity";
  "Krasnojarsk";
}
"Kuala_Lumpur";
{
  "exemplarCity";
  "Kuala Lumpur";
}
"Kuching";
{
  "exemplarCity";
  "Kuching";
}
"Kuwait";
{
  "exemplarCity";
  "Kuwait";
}
"Macau";
{
```



```
        "exemplarCity";
        "Macao";
    }
    "Magadan";
    {
        "exemplarCity";
        "Magadan";
    }
    "Makassar";
    {
        "exemplarCity";
        "Makassar";
    }
    "Manila";
    {
        "exemplarCity";
        "Manila";
    }
    "Muscat";
    {
        "exemplarCity";
        "Maskat";
    }
    "Nicosia";
    {
        "exemplarCity";
        "Nikosia";
    }
    "Novokuznetsk";
    {
        "exemplarCity";
        "Nowokuznetsk";
    }
    "Novosibirsk";
    {
        "exemplarCity";
        "Nowosibirsk";
    }
    "Omsk";
    {
        "exemplarCity";
        "Omsk";
    }
    "Oral";
    {
        "exemplarCity";
        "Oral";
    }
    "Phnom_Penh";
    {
        "exemplarCity";
        "Phnom Penh";
    }
    "Pontianak";
    {
        "exemplarCity";
        "Pontianak";
    }
}
```

```
}
"Pyongyang";
{
  "exemplarCity";
  "Pjöngjang";
}
"Qatar";
{
  "exemplarCity";
  "Katar";
}
"Qyzylorda";
{
  "exemplarCity";
  "Qysylorda";
}
"Rangoon";
{
  "exemplarCity";
  "Rangun";
}
"Riyadh";
{
  "exemplarCity";
  "Riad";
}
"Saigon";
{
  "exemplarCity";
  "Ho-Chi-Minh-Stadt";
}
"Sakhalin";
{
  "exemplarCity";
  "Sachalin";
}
"Samarkand";
{
  "exemplarCity";
  "Samarkand";
}
"Seoul";
{
  "exemplarCity";
  "Seoul";
}
"Shanghai";
{
  "exemplarCity";
  "Shanghai";
}
"Singapore";
{
  "exemplarCity";
  "Singapur";
}
"Srednekolymsk";
```

```
{
    "exemplarCity";
    "Srednekolymsk";
}
"Taipei";
{
    "exemplarCity";
    "Taipeh";
}
"Tashkent";
{
    "exemplarCity";
    "Taschkent";
}
"Tbilisi";
{
    "exemplarCity";
    "Tiflis";
}
"Tehran";
{
    "exemplarCity";
    "Teheran";
}
"Thimphu";
{
    "exemplarCity";
    "Thimphu";
}
"Tokyo";
{
    "exemplarCity";
    "Tokio";
}
"Tomsk";
{
    "exemplarCity";
    "Tomsk";
}
"Ulaanbaatar";
{
    "exemplarCity";
    "Ulaanbaatar";
}
"Urumqi";
{
    "exemplarCity";
    "Ürümqi";
}
"Ust-Nera";
{
    "exemplarCity";
    "Ust-Nera";
}
"Vientiane";
{
    "exemplarCity";
```

```
        "Vientiane";
    }
    "Vladivostok";
    {
        "exemplarCity";
        "Wladiwostok";
    }
    "Yakutsk";
    {
        "exemplarCity";
        "Jakutsk";
    }
    "Yekaterinburg";
    {
        "exemplarCity";
        "Jekaterinburg";
    }
    "Yerevan";
    {
        "exemplarCity";
        "Eriwan";
    }
}
"Indian";
{
    "Antananarivo";
    {
        "exemplarCity";
        "Antananarivo";
    }
    "Chagos";
    {
        "exemplarCity";
        "Chagos";
    }
    "Christmas";
    {
        "exemplarCity";
        "Weihnachtsinsel";
    }
    "Cocos";
    {
        "exemplarCity";
        "Cocos";
    }
    "Comoro";
    {
        "exemplarCity";
        "Komoren";
    }
    "Kerguelen";
    {
        "exemplarCity";
        "Kerguelen";
    }
    "Mahe";
    {
```

```
        "exemplarCity";
        "Mahe";
    }
    "Maldives";
    {
        "exemplarCity";
        "Malediven";
    }
    "Mauritius";
    {
        "exemplarCity";
        "Mauritius";
    }
    "Mayotte";
    {
        "exemplarCity";
        "Mayotte";
    }
    "Reunion";
    {
        "exemplarCity";
        "Réunion";
    }
}
"Australia";
{
    "Adelaide";
    {
        "exemplarCity";
        "Adelaide";
    }
    "Brisbane";
    {
        "exemplarCity";
        "Brisbane";
    }
    "Broken_Hill";
    {
        "exemplarCity";
        "Broken Hill";
    }
    "Currie";
    {
        "exemplarCity";
        "Currie";
    }
    "Darwin";
    {
        "exemplarCity";
        "Darwin";
    }
    "Eucla";
    {
        "exemplarCity";
        "Eucla";
    }
    "Hobart";
```

```
{
    "exemplarCity";
    "Hobart";
}
"Lindeman";
{
    "exemplarCity";
    "Lindeman";
}
"Lord_Howe";
{
    "exemplarCity";
    "Lord Howe";
}
"Melbourne";
{
    "exemplarCity";
    "Melbourne";
}
"Perth";
{
    "exemplarCity";
    "Perth";
}
"Sydney";
{
    "exemplarCity";
    "Sydney";
}
}
"Pacific";
{
    "Apia";
    {
        "exemplarCity";
        "Apia";
    }
    "Auckland";
    {
        "exemplarCity";
        "Auckland";
    }
    "Bougainville";
    {
        "exemplarCity";
        "Bougainville";
    }
    "Chatham";
    {
        "exemplarCity";
        "Chatham";
    }
    "Easter";
    {
        "exemplarCity";
        "Osterinsel";
    }
}
```

```
"Efate";
{
  "exemplarCity";
  "Efate";
}
"Enderbury";
{
  "exemplarCity";
  "Enderbury";
}
"Fakaofo";
{
  "exemplarCity";
  "Fakaofo";
}
"Fiji";
{
  "exemplarCity";
  "Fidschi";
}
"Funafuti";
{
  "exemplarCity";
  "Funafuti";
}
"Galapagos";
{
  "exemplarCity";
  "Galapagos";
}
"Gambier";
{
  "exemplarCity";
  "Gambier";
}
"Guadalcanal";
{
  "exemplarCity";
  "Guadalcanal";
}
"Guam";
{
  "exemplarCity";
  "Guam";
}
"Honolulu";
{
  "exemplarCity";
  "Honolulu";
}
"Johnston";
{
  "exemplarCity";
  "Johnston";
}
"Kiritimati";
{
```

```
        "exemplarCity";
        "Kiritimati";
    }
    "Kosrae";
    {
        "exemplarCity";
        "Kosrae";
    }
    "Kwajalein";
    {
        "exemplarCity";
        "Kwajalein";
    }
    "Majuro";
    {
        "exemplarCity";
        "Majuro";
    }
    "Marquesas";
    {
        "exemplarCity";
        "Marquesas";
    }
    "Midway";
    {
        "exemplarCity";
        "Midway";
    }
    "Nauru";
    {
        "exemplarCity";
        "Nauru";
    }
    "Niue";
    {
        "exemplarCity";
        "Niue";
    }
    "Norfolk";
    {
        "exemplarCity";
        "Norfolk";
    }
    "Noumea";
    {
        "exemplarCity";
        "Noumea";
    }
    "Pago_Pago";
    {
        "exemplarCity";
        "Pago Pago";
    }
    "Palau";
    {
        "exemplarCity";
        "Palau";
    }
```



```
}
"Pitcairn";
{
  "exemplarCity";
  "Pitcairn";
}
"Ponape";
{
  "exemplarCity";
  "Pohnpei";
}
"Port_Moresby";
{
  "exemplarCity";
  "Port Moresby";
}
"Rarotonga";
{
  "exemplarCity";
  "Rarotonga";
}
"Saipan";
{
  "exemplarCity";
  "Saipan";
}
"Tahiti";
{
  "exemplarCity";
  "Tahiti";
}
"Tarawa";
{
  "exemplarCity";
  "Tarawa";
}
"Tongatapu";
{
  "exemplarCity";
  "Tongatapu";
}
"Truk";
{
  "exemplarCity";
  "Chuuk";
}
"Wake";
{
  "exemplarCity";
  "Wake";
}
"Wallis";
{
  "exemplarCity";
  "Wallis";
}
}
```

```
"Arctic";
{
  "Longyearbyen";
  {
    "exemplarCity";
    "Longyearbyen";
  }
}
"Antarctica";
{
  "Casey";
  {
    "exemplarCity";
    "Casey";
  }
  "Davis";
  {
    "exemplarCity";
    "Davis";
  }
  "DumontDUrville";
  {
    "exemplarCity";
    "Dumont d'Urville";
  }
  "Macquarie";
  {
    "exemplarCity";
    "Macquarie";
  }
  "Mawson";
  {
    "exemplarCity";
    "Mawson";
  }
  "McMurdo";
  {
    "exemplarCity";
    "McMurdo";
  }
  "Palmer";
  {
    "exemplarCity";
    "Palmer";
  }
  "Rothera";
  {
    "exemplarCity";
    "Rothera";
  }
  "Syowa";
  {
    "exemplarCity";
    "Syowa";
  }
  "Troll";
  {
```

```
        "exemplarCity";
        "Troll";
    }
    "Vostok";
    {
        "exemplarCity";
        "Wostok";
    }
}
"Etc";
{
    "GMT";
    {
        "exemplarCity";
        "GMT";
    }
    "GMT1";
    {
        "exemplarCity";
        "GMT+1";
    }
    "GMT10";
    {
        "exemplarCity";
        "GMT+10";
    }
    "GMT11";
    {
        "exemplarCity";
        "GMT+11";
    }
    "GMT12";
    {
        "exemplarCity";
        "GMT+12";
    }
    "GMT2";
    {
        "exemplarCity";
        "GMT+2";
    }
    "GMT3";
    {
        "exemplarCity";
        "GMT+3";
    }
    "GMT4";
    {
        "exemplarCity";
        "GMT+4";
    }
    "GMT5";
    {
        "exemplarCity";
        "GMT+5";
    }
    "GMT6";
```

```
{
    "exemplarCity";
    "GMT+6";
}
"GMT7";
{
    "exemplarCity";
    "GMT+7";
}
"GMT8";
{
    "exemplarCity";
    "GMT+8";
}
"GMT9";
{
    "exemplarCity";
    "GMT+9";
}
"GMT-1";
{
    "exemplarCity";
    "GMT-1";
}
"GMT-10";
{
    "exemplarCity";
    "GMT-10";
}
"GMT-11";
{
    "exemplarCity";
    "GMT-11";
}
"GMT-12";
{
    "exemplarCity";
    "GMT-12";
}
"GMT-13";
{
    "exemplarCity";
    "GMT-13";
}
"GMT-14";
{
    "exemplarCity";
    "GMT-14";
}
"GMT-2";
{
    "exemplarCity";
    "GMT-2";
}
"GMT-3";
{
    "exemplarCity";
```

```

        "GMT-3";
    }
    "GMT-4";
    {
        "exemplarCity";
        "GMT-4";
    }
    "GMT-5";
    {
        "exemplarCity";
        "GMT-5";
    }
    "GMT-6";
    {
        "exemplarCity";
        "GMT-6";
    }
    "GMT-7";
    {
        "exemplarCity";
        "GMT-7";
    }
    "GMT-8";
    {
        "exemplarCity";
        "GMT-8";
    }
    "GMT-9";
    {
        "exemplarCity";
        "GMT-9";
    }
    "Unknown";
    {
        "exemplarCity";
        "Unbekannt";
    }
    }
}
"metazone";
{
    "Acre";
    {
        "long";
        {
            "generic";
            "Acre-Zeit",
            "standard";
            "Acre-Normalzeit",
            "daylight";
            "Acre-Sommerzeit";
        }
    }
    "Afghanistan";
    {
        "long";
        {

```

```
        "standard";
        "Afghanistan-Zeit";
    }
}
"Africa_Central";
{
    "long";
    {
        "standard";
        "Zentralafrikanische Zeit";
    }
}
"Africa_Eastern";
{
    "long";
    {
        "standard";
        "Ostafrikanische Zeit";
    }
}
"Africa_Southern";
{
    "long";
    {
        "standard";
        "Südafrikanische Zeit";
    }
}
"Africa_Western";
{
    "long";
    {
        "generic";
        "Westafrikanische Zeit",
        "standard";
        "Westafrikanische Normalzeit",
        "daylight";
        "Westafrikanische Sommerzeit";
    }
}
"Alaska";
{
    "long";
    {
        "generic";
        "Alaska-Zeit",
        "standard";
        "Alaska-Normalzeit",
        "daylight";
        "Alaska-Sommerzeit";
    }
}
"Almaty";
{
    "long";
    {
        "generic";
```

```

        "Almaty-Zeit",
        "standard";
        "Almaty-Normalzeit",
        "daylight";
        "Almaty-Sommerzeit";
    }
}
"Amazon";
{
    "long";
    {
        "generic";
        "Amazonas-Zeit",
        "standard";
        "Amazonas-Normalzeit",
        "daylight";
        "Amazonas-Sommerzeit";
    }
}
"America_Central";
{
    "long";
    {
        "generic";
        "Nordamerikanische Inlandzeit",
        "standard";
        "Nordamerikanische Inland-Normalzeit",
        "daylight";
        "Nordamerikanische Inland-Sommerzeit";
    }
}
"America_Eastern";
{
    "long";
    {
        "generic";
        "Nordamerikanische Ostküstenzeit",
        "standard";
        "Nordamerikanische Ostküsten-Normalzeit",
        "daylight";
        "Nordamerikanische Ostküsten-Sommerzeit";
    }
}
"America_Mountain";
{
    "long";
    {
        "generic";
        "Rocky-Mountain-Zeit",
        "standard";
        "Rocky Mountain-Normalzeit",
        "daylight";
        "Rocky-Mountain-Sommerzeit";
    }
}
"America_Pacific";
{

```

```
        "long";
        {
            "generic";
            "Nordamerikanische Westküstenzeit",
            "standard";
            "Nordamerikanische Westküsten-Normalzeit",
            "daylight";
            "Nordamerikanische Westküsten-Sommerzeit";
        }
    }
    "Anadyr";
    {
        "long";
        {
            "generic";
            "Anadyr Zeit",
            "standard";
            "Anadyr Normalzeit",
            "daylight";
            "Anadyr Sommerzeit";
        }
    }
    "Apia";
    {
        "long";
        {
            "generic";
            "Apia-Zeit",
            "standard";
            "Apia-Normalzeit",
            "daylight";
            "Apia-Sommerzeit";
        }
    }
    "Aqtau";
    {
        "long";
        {
            "generic";
            "Aqtau-Zeit",
            "standard";
            "Aqtau-Normalzeit",
            "daylight";
            "Aqtau-Sommerzeit";
        }
    }
    "Aqtobe";
    {
        "long";
        {
            "generic";
            "Aqtöbe-Zeit",
            "standard";
            "Aqtöbe-Normalzeit",
            "daylight";
            "Aqtöbe-Sommerzeit";
        }
    }
}
```



```
}
"Arabic";
{
  "long";
  {
    "generic";
    "Arabisches Zeit",
    "standard";
    "Arabisches Normalzeit",
    "daylight";
    "Arabisches Sommerzeit";
  }
}
"Argentina";
{
  "long";
  {
    "generic";
    "Argentinische Zeit",
    "standard";
    "Argentinische Normalzeit",
    "daylight";
    "Argentinische Sommerzeit";
  }
}
"Argentina_Western";
{
  "long";
  {
    "generic";
    "Westargentinische Zeit",
    "standard";
    "Westargentinische Normalzeit",
    "daylight";
    "Westargentinische Sommerzeit";
  }
}
"Armenia";
{
  "long";
  {
    "generic";
    "Armenische Zeit",
    "standard";
    "Armenische Normalzeit",
    "daylight";
    "Armenische Sommerzeit";
  }
}
"Atlantic";
{
  "long";
  {
    "generic";
    "Atlantik-Zeit",
    "standard";
    "Atlantik-Normalzeit",
```

```

        "daylight";
        "Atlantik-Sommerzeit";
    }
}
"Australia_Central";
{
    "long";
    {
        "generic";
        "Zentralaustralische Zeit",
        "standard";
        "Zentralaustralische Normalzeit",
        "daylight";
        "Zentralaustralische Sommerzeit";
    }
}
"Australia_CentralWestern";
{
    "long";
    {
        "generic";
        "Zentral-/Westaustralische Zeit",
        "standard";
        "Zentral-/Westaustralische Normalzeit",
        "daylight";
        "Zentral-/Westaustralische Sommerzeit";
    }
}
"Australia_Eastern";
{
    "long";
    {
        "generic";
        "Ostaustralische Zeit",
        "standard";
        "Ostaustralische Normalzeit",
        "daylight";
        "Ostaustralische Sommerzeit";
    }
}
"Australia_Western";
{
    "long";
    {
        "generic";
        "Westaustralische Zeit",
        "standard";
        "Westaustralische Normalzeit",
        "daylight";
        "Westaustralische Sommerzeit";
    }
}
"Azerbaijan";
{
    "long";
    {
        "generic";

```

```

        "Aserbaidtschanische Zeit",
        "standard";
        "Aserbeidschanische Normalzeit",
        "daylight";
        "Aserbaidtschanische Sommerzeit";
    }
}
"Azores";
{
    "long";
    {
        "generic";
        "Azoren-Zeit",
        "standard";
        "Azoren-Normalzeit",
        "daylight";
        "Azoren-Sommerzeit";
    }
}
"Bangladesh";
{
    "long";
    {
        "generic";
        "Bangladesch-Zeit",
        "standard";
        "Bangladesch-Normalzeit",
        "daylight";
        "Bangladesch-Sommerzeit";
    }
}
"Bhutan";
{
    "long";
    {
        "standard";
        "Bhutan-Zeit";
    }
}
"Bolivia";
{
    "long";
    {
        "standard";
        "Bolivianische Zeit";
    }
}
"Brasilia";
{
    "long";
    {
        "generic";
        "Brasília-Zeit",
        "standard";
        "Brasília-Normalzeit",
        "daylight";
        "Brasília-Sommerzeit";
    }
}

```

```
    }  
  }  
  "Brunei";  
  {  
    "long";  
    {  
      "standard";  
      "Brunei-Zeit";  
    }  
  }  
  "Cape_Verde";  
  {  
    "long";  
    {  
      "generic";  
      "Cabo-Verde-Zeit",  
      "standard";  
      "Cabo-Verde-Normalzeit",  
      "daylight";  
      "Cabo-Verde-Sommerzeit";  
    }  
  }  
  "Casey";  
  {  
    "long";  
    {  
      "standard";  
      "Casey-Zeit";  
    }  
  }  
  "Chamorro";  
  {  
    "long";  
    {  
      "standard";  
      "Chamorro-Zeit";  
    }  
  }  
  "Chatham";  
  {  
    "long";  
    {  
      "generic";  
      "Chatham-Zeit",  
      "standard";  
      "Chatham-Normalzeit",  
      "daylight";  
      "Chatham-Sommerzeit";  
    }  
  }  
  "Chile";  
  {  
    "long";  
    {  
      "generic";  
      "Chilenische Zeit",  
      "standard";  
    }  
  }
```

```
        "Chilenische Normalzeit",
        "daylight";
        "Chilenische Sommerzeit";
    }
}
"China";
{
    "long";
    {
        "generic";
        "Chinesische Zeit",
        "standard";
        "Chinesische Normalzeit",
        "daylight";
        "Chinesische Sommerzeit";
    }
}
"Choibalsan";
{
    "long";
    {
        "generic";
        "Tschoibalsan-Zeit",
        "standard";
        "Tschoibalsan-Normalzeit",
        "daylight";
        "Tschoibalsan-Sommerzeit";
    }
}
"Christmas";
{
    "long";
    {
        "standard";
        "Weihnachtsinsel-Zeit";
    }
}
"Cocos";
{
    "long";
    {
        "standard";
        "Kokosinseln-Zeit";
    }
}
"Colombia";
{
    "long";
    {
        "generic";
        "Kolumbianische Zeit",
        "standard";
        "Kolumbianische Normalzeit",
        "daylight";
        "Kolumbianische Sommerzeit";
    }
}
}
```

```
"Cook";
{
  "long";
  {
    "generic";
    "Cookinseln-Zeit",
    "standard";
    "Cookinseln-Normalzeit",
    "daylight";
    "Cookinseln-Sommerzeit";
  }
}
"Cuba";
{
  "long";
  {
    "generic";
    "Kubanische Zeit",
    "standard";
    "Kubanische Normalzeit",
    "daylight";
    "Kubanische Sommerzeit";
  }
}
"Davis";
{
  "long";
  {
    "standard";
    "Davis-Zeit";
  }
}
"DumontDUrville";
{
  "long";
  {
    "standard";
    "Dumont-d'Urville-Zeit";
  }
}
"East_Timor";
{
  "long";
  {
    "standard";
    "Osttimor-Zeit";
  }
}
"Easter";
{
  "long";
  {
    "generic";
    "Osterinsel-Zeit",
    "standard";
    "Osterinsel-Normalzeit",
    "daylight";
  }
}
```

```

        "Osterinsel-Sommerzeit";
    }
}
"Ecuador";
{
    "long";
    {
        "standard";
        "Ecuadorianische Zeit";
    }
}
"Europe_Central";
{
    "long";
    {
        "generic";
        "Mittleuropäische Zeit",
        "standard";
        "Mittleuropäische Normalzeit",
        "daylight";
        "Mittleuropäische Sommerzeit";
    }
    "short";
    {
        "generic";
        "MEZ",
        "standard";
        "MEZ",
        "daylight";
        "MESZ";
    }
}
"Europe_Eastern";
{
    "long";
    {
        "generic";
        "Osteuropäische Zeit",
        "standard";
        "Osteuropäische Normalzeit",
        "daylight";
        "Osteuropäische Sommerzeit";
    }
    "short";
    {
        "generic";
        "OEZ",
        "standard";
        "OEZ",
        "daylight";
        "OESZ";
    }
}
"Europe_Further_Eastern";
{
    "long";
    {

```

```
        "standard";
        "Kaliningrader Zeit";
    }
}
"Europe_Western";
{
    "long";
    {
        "generic";
        "Westeuropäische Zeit",
        "standard";
        "Westeuropäische Normalzeit",
        "daylight";
        "Westeuropäische Sommerzeit";
    }
    "short";
    {
        "generic";
        "WEZ",
        "standard";
        "WEZ",
        "daylight";
        "WESZ";
    }
}
"Falkland";
{
    "long";
    {
        "generic";
        "Falklandinseln-Zeit",
        "standard";
        "Falklandinseln-Normalzeit",
        "daylight";
        "Falklandinseln-Sommerzeit";
    }
}
"Fiji";
{
    "long";
    {
        "generic";
        "Fidschi-Zeit",
        "standard";
        "Fidschi-Normalzeit",
        "daylight";
        "Fidschi-Sommerzeit";
    }
}
"French_Guiana";
{
    "long";
    {
        "standard";
        "Französisch-Guayana-Zeit";
    }
}
```



```

    "French_Southern";
    {
        "long";
        {
            "standard";
            "Französische Süd- und Antarktisgebiete-
Zeit";
        }
    }
    "Galapagos";
    {
        "long";
        {
            "standard";
            "Galapagos-Zeit";
        }
    }
    "Gambier";
    {
        "long";
        {
            "standard";
            "Gambier-Zeit";
        }
    }
    "Georgia";
    {
        "long";
        {
            "generic";
            "Georgische Zeit",
            "standard";
            "Georgische Normalzeit",
            "daylight";
            "Georgische Sommerzeit";
        }
    }
    "Gilbert_Islands";
    {
        "long";
        {
            "standard";
            "Gilbert-Inseln-Zeit";
        }
    }
    "GMT";
    {
        "long";
        {
            "standard";
            "Mittlere Greenwich-Zeit";
        }
    }
    "Greenland_Eastern";
    {
        "long";
        {

```

```

        "generic";
        "Ostgrönland-Zeit",
        "standard";
        "Ostgrönland-Normalzeit",
        "daylight";
        "Ostgrönland-Sommerzeit";
    }
}
"Greenland_Western";
{
    "long";
    {
        "generic";
        "Westgrönland-Zeit",
        "standard";
        "Westgrönland-Normalzeit",
        "daylight";
        "Westgrönland-Sommerzeit";
    }
}
"Guam";
{
    "long";
    {
        "standard";
        "Guam-Zeit";
    }
}
"Gulf";
{
    "long";
    {
        "standard";
        "Golf-Zeit";
    }
}
"Guyana";
{
    "long";
    {
        "standard";
        "Guyana-Zeit";
    }
}
"Hawaii_Aleutian";
{
    "long";
    {
        "generic";
        "Hawaii-Aleuten-Zeit",
        "standard";
        "Hawaii-Aleuten-Normalzeit",
        "daylight";
        "Hawaii-Aleuten-Sommerzeit";
    }
}
"Hong_Kong";

```

```
{
  "long";
  {
    "generic";
    "Hongkong-Zeit",
    "standard";
    "Hongkong-Normalzeit",
    "daylight";
    "Hongkong-Sommerzeit";
  }
}
"Hovd";
{
  "long";
  {
    "generic";
    "Chowd-Zeit",
    "standard";
    "Chowd-Normalzeit",
    "daylight";
    "Chowd-Sommerzeit";
  }
}
"India";
{
  "long";
  {
    "standard";
    "Indische Zeit";
  }
}
"Indian_Ocean";
{
  "long";
  {
    "standard";
    "Indischer Ozean-Zeit";
  }
}
"Indochina";
{
  "long";
  {
    "standard";
    "Indochina-Zeit";
  }
}
"Indonesia_Central";
{
  "long";
  {
    "standard";
    "Zentralindonesische Zeit";
  }
}
"Indonesia_Eastern";
{
```

```
        "long";
        {
            "standard";
            "Ostindonesische Zeit";
        }
    }
    "Indonesia_Western";
    {
        "long";
        {
            "standard";
            "Westindonesische Zeit";
        }
    }
    "Iran";
    {
        "long";
        {
            "generic";
            "Iranische Zeit",
            "standard";
            "Iranische Normalzeit",
            "daylight";
            "Iranische Sommerzeit";
        }
    }
    "Irkutsk";
    {
        "long";
        {
            "generic";
            "Irkutsk-Zeit",
            "standard";
            "Irkutsk-Normalzeit",
            "daylight";
            "Irkutsk-Sommerzeit";
        }
    }
    "Israel";
    {
        "long";
        {
            "generic";
            "Israelische Zeit",
            "standard";
            "Israelische Normalzeit",
            "daylight";
            "Israelische Sommerzeit";
        }
    }
    "Japan";
    {
        "long";
        {
            "generic";
            "Japanische Zeit",
            "standard";
```

```

        "Japanische Normalzeit",
        "daylight";
        "Japanische Sommerzeit";
    }
}
"Kamchatka";
{
    "long";
    {
        "generic";
        "Kamtschatka-Zeit",
        "standard";
        "Kamtschatka-Normalzeit",
        "daylight";
        "Kamtschatka-Sommerzeit";
    }
}
"Kazakhstan_Eastern";
{
    "long";
    {
        "standard";
        "Ostkasachische Zeit";
    }
}
"Kazakhstan_Western";
{
    "long";
    {
        "standard";
        "Westkasachische Zeit";
    }
}
"Korea";
{
    "long";
    {
        "generic";
        "Koreanische Zeit",
        "standard";
        "Koreanische Normalzeit",
        "daylight";
        "Koreanische Sommerzeit";
    }
}
"Kosrae";
{
    "long";
    {
        "standard";
        "Kosrae-Zeit";
    }
}
"Krasnoyarsk";
{
    "long";
    {

```

```
        "generic";
        "Krasnojarsk-Zeit",
        "standard";
        "Krasnojarsk-Normalzeit",
        "daylight";
        "Krasnojarsk-Sommerzeit";
    }
}
"Kyrgystan";
{
    "long";
    {
        "standard";
        "Kirgisistan-Zeit";
    }
}
"Lanka";
{
    "long";
    {
        "standard";
        "Sri-Lanka-Zeit";
    }
}
"Line_Islands";
{
    "long";
    {
        "standard";
        "Linieninseln-Zeit";
    }
}
"Lord_Howe";
{
    "long";
    {
        "generic";
        "Lord-Howe-Zeit",
        "standard";
        "Lord-Howe-Normalzeit",
        "daylight";
        "Lord-Howe-Sommerzeit";
    }
}
"Macau";
{
    "long";
    {
        "generic";
        "Macau-Zeit",
        "standard";
        "Macau-Normalzeit",
        "daylight";
        "Macau-Sommerzeit";
    }
}
"Macquarie";
```

```

    {
      "long";
      {
        "standard";
        "Macquarieinsel-Zeit";
      }
    }
    "Magadan";
    {
      "long";
      {
        "generic";
        "Magadan-Zeit",
        "standard";
        "Magadan-Normalzeit",
        "daylight";
        "Magadan-Sommerzeit";
      }
    }
    "Malaysia";
    {
      "long";
      {
        "standard";
        "Malaysische Zeit";
      }
    }
    "Maldives";
    {
      "long";
      {
        "standard";
        "Malediven-Zeit";
      }
    }
    "Marquesas";
    {
      "long";
      {
        "standard";
        "Marquesas-Zeit";
      }
    }
    "Marshall_Islands";
    {
      "long";
      {
        "standard";
        "Marshallinseln-Zeit";
      }
    }
    "Mauritius";
    {
      "long";
      {
        "generic";
        "Mauritius-Zeit",

```

```

        "standard";
        "Mauritius-Normalzeit",
        "daylight";
        "Mauritius-Sommerzeit";
    }
}
"Mawson";
{
    "long";
    {
        "standard";
        "Mawson-Zeit";
    }
}
"Mexico_Northwest";
{
    "long";
    {
        "generic";
        "Mexiko Nordwestliche Zone-Zeit",
        "standard";
        "Mexiko Nordwestliche Zone-Normalzeit",
        "daylight";
        "Mexiko Nordwestliche Zone-Sommerzeit";
    }
}
"Mexico_Pacific";
{
    "long";
    {
        "generic";
        "Mexiko Pazifikzone-Zeit",
        "standard";
        "Mexiko Pazifikzone-Normalzeit",
        "daylight";
        "Mexiko Pazifikzone-Sommerzeit";
    }
}
"Mongolia";
{
    "long";
    {
        "generic";
        "Ulaanbaatar-Zeit",
        "standard";
        "Ulaanbaatar-Normalzeit",
        "daylight";
        "Ulaanbaatar-Sommerzeit";
    }
}
"Moscow";
{
    "long";
    {
        "generic";
        "Moskauer Zeit",
        "standard";
    }
}

```



```
        "Moskauer Normalzeit",
        "daylight";
        "Moskauer Sommerzeit";
    }
}
"Myanmar";
{
    "long";
    {
        "standard";
        "Myanmar-Zeit";
    }
}
"Nauru";
{
    "long";
    {
        "standard";
        "Nauru-Zeit";
    }
}
"Nepal";
{
    "long";
    {
        "standard";
        "Nepalesische Zeit";
    }
}
"New_Caledonia";
{
    "long";
    {
        "generic";
        "Neukaledonische Zeit",
        "standard";
        "Neukaledonische Normalzeit",
        "daylight";
        "Neukaledonische Sommerzeit";
    }
}
"New_Zealand";
{
    "long";
    {
        "generic";
        "Neuseeland-Zeit",
        "standard";
        "Neuseeland-Normalzeit",
        "daylight";
        "Neuseeland-Sommerzeit";
    }
}
"Newfoundland";
{
    "long";
    {
```

```
        "generic";
        "Neufundland-Zeit",
        "standard";
        "Neufundland-Normalzeit",
        "daylight";
        "Neufundland-Sommerzeit";
    }
}
"Niue";
{
    "long";
    {
        "standard";
        "Niue-Zeit";
    }
}
"Norfolk";
{
    "long";
    {
        "standard";
        "Norfolkinsel-Zeit";
    }
}
"Noronha";
{
    "long";
    {
        "generic";
        "Fernando de Noronha-Zeit",
        "standard";
        "Fernando de Noronha-Normalzeit",
        "daylight";
        "Fernando de Noronha-Sommerzeit";
    }
}
"North_Mariana";
{
    "long";
    {
        "standard";
        "Nördliche-Marianen-Zeit";
    }
}
"Novosibirsk";
{
    "long";
    {
        "generic";
        "Nowosibirsk-Zeit",
        "standard";
        "Nowosibirsk-Normalzeit",
        "daylight";
        "Nowosibirsk-Sommerzeit";
    }
}
"Omsk";
```

```
{
  "long";
  {
    "generic";
    "Omsk-Zeit",
    "standard";
    "Omsk-Normalzeit",
    "daylight";
    "Omsk-Sommerzeit";
  }
}
"Pakistan";
{
  "long";
  {
    "generic";
    "Pakistanische Zeit",
    "standard";
    "Pakistanische Normalzeit",
    "daylight";
    "Pakistanische Sommerzeit";
  }
}
"Palau";
{
  "long";
  {
    "standard";
    "Palau-Zeit";
  }
}
"Papua_New_Guinea";
{
  "long";
  {
    "standard";
    "Papua-Neuguinea-Zeit";
  }
}
"Paraguay";
{
  "long";
  {
    "generic";
    "Paraguayische Zeit",
    "standard";
    "Paraguayische Normalzeit",
    "daylight";
    "Paraguayische Sommerzeit";
  }
}
"Peru";
{
  "long";
  {
    "generic";
    "Peruanische Zeit",
```

```

        "standard";
        "Peruanische Normalzeit",
        "daylight";
        "Peruanische Sommerzeit";
    }
}
"Philippines";
{
    "long";
    {
        "generic";
        "Philippinische Zeit",
        "standard";
        "Philippinische Normalzeit",
        "daylight";
        "Philippinische Sommerzeit";
    }
}
"Phoenix_Islands";
{
    "long";
    {
        "standard";
        "Phoenixinseln-Zeit";
    }
}
"Pierre_Miquelon";
{
    "long";
    {
        "generic";
        "Saint-Pierre-und-Miquelon-Zeit",
        "standard";
        "Saint-Pierre-und-Miquelon-Normalzeit",
        "daylight";
        "Saint-Pierre-und-Miquelon-Sommerzeit";
    }
}
"Pitcairn";
{
    "long";
    {
        "standard";
        "Pitcairnsinseln-Zeit";
    }
}
"Ponape";
{
    "long";
    {
        "standard";
        "Ponape-Zeit";
    }
}
"Pyongyang";
{
    "long";

```

```
        {
            "standard";
            "Pjöngjang-Zeit";
        }
    }
    "Qyzylorda";
    {
        "long";
        {
            "generic";
            "Quysylorda-Zeit",
            "standard";
            "Quysylorda-Normalzeit",
            "daylight";
            "Qysylorda-Sommerzeit";
        }
    }
    "Reunion";
    {
        "long";
        {
            "standard";
            "Réunion-Zeit";
        }
    }
    "Rothera";
    {
        "long";
        {
            "standard";
            "Rothera-Zeit";
        }
    }
    "Sakhalin";
    {
        "long";
        {
            "generic";
            "Sachalin-Zeit",
            "standard";
            "Sachalin-Normalzeit",
            "daylight";
            "Sachalin-Sommerzeit";
        }
    }
    "Samara";
    {
        "long";
        {
            "generic";
            "Samara-Zeit",
            "standard";
            "Samara-Normalzeit",
            "daylight";
            "Samara-Sommerzeit";
        }
    }
}
```

```
"Samoa";
{
  "long";
  {
    "generic";
    "Samoa-Zeit",
    "standard";
    "Samoa-Normalzeit",
    "daylight";
    "Samoa-Sommerzeit";
  }
}
"Seychelles";
{
  "long";
  {
    "standard";
    "Seychellen-Zeit";
  }
}
"Singapore";
{
  "long";
  {
    "standard";
    "Singapur-Zeit";
  }
}
"Solomon";
{
  "long";
  {
    "standard";
    "Salomoninseln-Zeit";
  }
}
"South_Georgia";
{
  "long";
  {
    "standard";
    "Südgeorgische Zeit";
  }
}
"Suriname";
{
  "long";
  {
    "standard";
    "Suriname-Zeit";
  }
}
"Syowa";
{
  "long";
  {
    "standard";
```

```

        "Syowa-Zeit";
    }
}
"Tahiti";
{
    "long";
    {
        "standard";
        "Tahiti-Zeit";
    }
}
"Taipei";
{
    "long";
    {
        "generic";
        "Taipeh-Zeit",
        "standard";
        "Taipeh-Normalzeit",
        "daylight";
        "Taipeh-Sommerzeit";
    }
}
"Tajikistan";
{
    "long";
    {
        "standard";
        "Tadschikistan-Zeit";
    }
}
"Tokelau";
{
    "long";
    {
        "standard";
        "Tokelau-Zeit";
    }
}
"Tonga";
{
    "long";
    {
        "generic";
        "Tonganische Zeit",
        "standard";
        "Tonganische Normalzeit",
        "daylight";
        "Tonganische Sommerzeit";
    }
}
"Truk";
{
    "long";
    {
        "standard";
        "Chuuk-Zeit";
    }
}

```

```
    }  
  }  
  "Turkmenistan";  
  {  
    "long";  
    {  
      "generic";  
      "Turkmenistan-Zeit",  
        "standard";  
      "Turkmenistan-Normalzeit",  
        "daylight";  
      "Turkmenistan-Sommerzeit";  
    }  
  }  
  "Tuvalu";  
  {  
    "long";  
    {  
      "standard";  
      "Tuvalu-Zeit";  
    }  
  }  
  "Uruguay";  
  {  
    "long";  
    {  
      "generic";  
      "Uruguayische Zeit",  
        "standard";  
      "Uruguayische Normalzeit",  
        "daylight";  
      "Uruguayische Sommerzeit";  
    }  
  }  
  "Uzbekistan";  
  {  
    "long";  
    {  
      "generic";  
      "Usbekistan-Zeit",  
        "standard";  
      "Usbekistan-Normalzeit",  
        "daylight";  
      "Usbekistan-Sommerzeit";  
    }  
  }  
  "Vanuatu";  
  {  
    "long";  
    {  
      "generic";  
      "Vanuatu-Zeit",  
        "standard";  
      "Vanuatu-Normalzeit",  
        "daylight";  
      "Vanuatu-Sommerzeit";  
    }  
  }  
}
```



```
}
"Venezuela";
{
  "long";
  {
    "standard";
    "Venezuela-Zeit";
  }
}
"Vladivostok";
{
  "long";
  {
    "generic";
    "Wladiwostok-Zeit",
    "standard";
    "Wladiwostok-Normalzeit",
    "daylight";
    "Wladiwostok-Sommerzeit";
  }
}
"Volgograd";
{
  "long";
  {
    "generic";
    "Wolgograd-Zeit",
    "standard";
    "Wolgograd-Normalzeit",
    "daylight";
    "Wolgograd-Sommerzeit";
  }
}
"Vostok";
{
  "long";
  {
    "standard";
    "Wostok-Zeit";
  }
}
"Wake";
{
  "long";
  {
    "standard";
    "Wake-Insel-Zeit";
  }
}
"Wallis";
{
  "long";
  {
    "standard";
    "Wallis-und-Futuna-Zeit";
  }
}
}
```

```

        "Yakutsk";
        {
            "long";
            {
                "generic";
                "Jakutsk-Zeit",
                "standard";
                "Jakutsk-Normalzeit",
                "daylight";
                "Jakutsk-Sommerzeit";
            }
        }
        "Yekaterinburg";
        {
            "long";
            {
                "generic";
                "Jekaterinburg-Zeit",
                "standard";
                "Jekaterinburg-Normalzeit",
                "daylight";
                "Jekaterinburg-Sommerzeit";
            }
        }
    }
}

```

Right-To-Left

The DateRangePicker supports right-to-left functionality for languages like Arabic, Hebrew, etc. To display the text in the right-to-left direction, use [enableRtl](#) property.

The following code example demonstrates the DateRangePicker component in **Hebrew** culture and also explains how to set the localized text to the placeholder using [load](#) method of [L10n](#) class.

[Class-component]

CA-GREGORIAN.JSON

```

{
  "main": {
    "he": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31.0.1"
        },
        "language": "he"
      },
      "dates": {
        "calendars": {
          "gregorian": {
            "months": {

```

```

    "format": {
      "abbreviated": {
        "1": "ינו'",
        "2": "פבר'",
        "3": "מרץ",
        "4": "אפר'",
        "5": "מאי",
        "6": "יוני",
        "7": "יולי",
        "8": "אוג'",
        "9": "ספט'",
        "10": "אוק'",
        "11": "נוב'",
        "12": "דצמ'"
      },
      "narrow": {
        "1": "1",
        "2": "2",
        "3": "3",
        "4": "4",
        "5": "5",
        "6": "6",
        "7": "7",
        "8": "8",
        "9": "9",
        "10": "10",
        "11": "11",
        "12": "12"
      },
      "wide": {
        "1": "ינואר",
        "2": "פברואר",
        "3": "מרץ",
        "4": "אפריל",
        "5": "מאי",
        "6": "יוני",
        "7": "יולי",
        "8": "אוגוסט",
        "9": "ספטמבר",
        "10": "אוקטובר",
        "11": "נובמבר",
        "12": "דצמבר"
      }
    },
    "stand-alone": {
      "abbreviated": {
        "1": "ינו'",
        "2": "פבר'",
        "3": "מרץ",
        "4": "אפר'",
        "5": "מאי",
        "6": "יוני",
        "7": "יולי",
        "8": "אוג'",
        "9": "ספט'",
        "10": "אוק'",
        "11": "נוב'",

```

```

        "12": "דצמב'ר",
      },
      "narrow": {
        "1": "1",
        "2": "2",
        "3": "3",
        "4": "4",
        "5": "5",
        "6": "6",
        "7": "7",
        "8": "8",
        "9": "9",
        "10": "10",
        "11": "11",
        "12": "12"
      },
      "wide": {
        "1": "ינואר",
        "2": "פברואר",
        "3": "מרץ",
        "4": "אפריל",
        "5": "מאי",
        "6": "יוני",
        "7": "יולי",
        "8": "אוגוסט",
        "9": "ספטמבר",
        "10": "אוקטובר",
        "11": "נובמבר",
        "12": "דצמבר"
      }
    },
    "days": {
      "format": {
        "abbreviated": {
          "sun": "יום א'",
          "mon": "יום ב'",
          "tue": "יום ג'",
          "wed": "יום ד'",
          "thu": "יום ה'",
          "fri": "יום ו'",
          "sat": "שבת"
        },
        "narrow": {
          "sun": "א'",
          "mon": "ב'",
          "tue": "ג'",
          "wed": "ד'",
          "thu": "ה'",
          "fri": "ו'",
          "sat": "ש"
        },
        "short": {
          "sun": "א",
          "mon": "ב",
          "tue": "ג",
          "wed": "ד",

```

```

        "thu": "ה'",
        "fri": "ו'",
        "sat": "ש'"},
    },
    "wide": {
      "sun": "יום ראשון",
      "mon": "יום שני",
      "tue": "יום שלישי",
      "wed": "יום רביעי",
      "thu": "יום חמישי",
      "fri": "יום שישי",
      "sat": "יום שבת"
    }
  },
  "stand-alone": {
    "abbreviated": {
      "sun": "א'",
      "mon": "ב'",
      "tue": "ג'",
      "wed": "ד'",
      "thu": "ה'",
      "fri": "ו'",
      "sat": "שבת"
    },
    "narrow": {
      "sun": "א",
      "mon": "ב",
      "tue": "ג",
      "wed": "ד",
      "thu": "ה",
      "fri": "ו",
      "sat": "ש"
    },
    "short": {
      "sun": "א",
      "mon": "ב",
      "tue": "ג",
      "wed": "ד",
      "thu": "ה",
      "fri": "ו",
      "sat": "ש"
    },
    "wide": {
      "sun": "יום ראשון",
      "mon": "יום שני",
      "tue": "יום שלישי",
      "wed": "יום רביעי",
      "thu": "יום חמישי",
      "fri": "יום שישי",
      "sat": "יום שבת"
    }
  }
},
"quarters": {
  "format": {
    "abbreviated": {
      "1": "Q1",

```

```

        "2": "Q2",
        "3": "Q3",
        "4": "Q4"
    },
    "narrow": {
        "1": "1",
        "2": "2",
        "3": "3",
        "4": "4"
    },
    "wide": {
        "1": "1 רבעון",
        "2": "2 רבעון",
        "3": "3 רבעון",
        "4": "4 רבעון"
    }
},
"stand-alone": {
    "abbreviated": {
        "1": "Q1",
        "2": "Q2",
        "3": "Q3",
        "4": "Q4"
    },
    "narrow": {
        "1": "1",
        "2": "2",
        "3": "3",
        "4": "4"
    },
    "wide": {
        "1": "1 רבעון",
        "2": "2 רבעון",
        "3": "3 רבעון",
        "4": "4 רבעון"
    }
}
},
"dayPeriods": {
    "format": {
        "abbreviated": {
            "midnight": "חצות",
            "am": "לפנה״צ",
            "pm": "אחה״צ",
            "morning1": "בוקר",
            "afternoon1": "צהריים",
            "afternoon2": "אחר הצהריים",
            "evening1": "ערב",
            "night1": "לילה",
            "night2": "לפנות בוקר"
        },
        "narrow": {
            "midnight": "חצות",
            "am": "לפנה״צ",
            "pm": "אחה״צ",
            "morning1": "בוקר",
            "afternoon1": "צהריים",

```

```

        "afternoon2": "אחר הצהריים",
        "evening1": "ערב",
        "night1": "לילה",
        "night2": "לפנות בוקר"
    },
    "wide": {
        "midnight": "חצות",
        "am": "לפנה"צ",
        "pm": "אחה"צ",
        "morning1": "בוקר",
        "afternoon1": "צהריים",
        "afternoon2": "אחר הצהריים",
        "evening1": "ערב",
        "night1": "לילה",
        "night2": "לפנות בוקר"
    }
},
"stand-alone": {
    "abbreviated": {
        "midnight": "חצות",
        "am": "לפנה"צ",
        "pm": "אחה"צ",
        "morning1": "בוקר",
        "afternoon1": "צהריים",
        "afternoon2": "אחר הצהריים",
        "evening1": "ערב",
        "night1": "לילה",
        "night2": "לפנות בוקר"
    },
    "narrow": {
        "midnight": "חצות",
        "am": "לפנה"צ",
        "pm": "אחה"צ",
        "morning1": "בוקר",
        "afternoon1": "צהריים",
        "afternoon2": "אחר הצהריים",
        "evening1": "ערב",
        "night1": "לילה",
        "night2": "לפנות בוקר"
    },
    "wide": {
        "midnight": "חצות",
        "am": "לפנה"צ",
        "pm": "אחה"צ",
        "morning1": "בוקר",
        "afternoon1": "צהריים",
        "afternoon2": "אחר הצהריים",
        "evening1": "ערב",
        "night1": "לילה",
        "night2": "לפנות בוקר"
    }
}
},
"eras": {
    "eraNames": {
        "0": "לפני הספירה",
        "0-alt-variant": "לפנה"ס",

```

```

        "1": "לטפירה",
        "1-alt-variant": "CE"
    },
    "eraAbbr": {
        "0": "לפנה"ס",
        "0-alt-variant": "BCE",
        "1": "לטפירה",
        "1-alt-variant": "CE"
    },
    "eraNarrow": {
        "0": "לפנה"ס",
        "0-alt-variant": "BCE",
        "1": "לטפירה",
        "1-alt-variant": "CE"
    }
},
"dateFormats": {
    "full": "EEEE, d מMMM y",
    "long": "d מMMM y",
    "medium": "d מMMM y",
    "short": "d.M.y"
},
"timeFormats": {
    "full": "H:mm:ss zzzz",
    "long": "H:mm:ss z",
    "medium": "H:mm:ss",
    "short": "H:mm"
},
"dateTimeFormats": {
    "full": "{1} בשעה {0}",
    "long": "{1} בשעה {0}",
    "medium": "{1}, {0}",
    "short": "{1}, {0}",
    "availableFormats": {
        "d": "d",
        "E": "ccc",
        "Ed": "E ה-d",
        "Ehm": "E h:mm a",
        "EHm": "E H:mm",
        "Ehms": "E h:mm:ss a",
        "EHms": "E H:mm:ss",
        "Gy": "y G",
        "GyMMM": "MMM y G",
        "GyMMMd": "d מMMM y G",
        "GyMMMED": "E, d מMMM y G",
        "h": "h a",
        "H": "H",
        "hm": "h:mm a",
        "Hm": "H:mm",
        "hms": "h:mm:ss a",
        "Hms": "H:mm:ss",
        "hmsv": "h:mm:ss a v",
        "Hmsv": "HH:mm:ss v",
        "hmv": "h:mm a v",
        "Hmv": "HH:mm v",
        "M": "L",
        "Md": "d.M",
    }
}

```



```

"MEd": "E, d.M",
"MMM": "LLL",
"MMMd": "d מMMM",
"MMMEd": "E, d מMMM",
"MMMMd": "d מMMMM",
"MMMMW-count-one": "שבוע W מMMM",
"MMMMW-count-two": "שבוע W מMMM",
"MMMMW-count-many": "שבוע W מMMM",
"MMMMW-count-other": "שבוע W מMMM",
"ms": "mm:ss",
"Y": "Y",
"yM": "M.Y",
"yMd": "d.M.Y",
"yMEd": "E, d.M.Y",
"yMM": "M.Y",
"yMMM": "MMM Y",
"yMMMd": "d מMMM Y",
"yMMMEd": "E, d מMMM Y",
"yMMMM": "MMMM Y",
"yQQQ": "QQQ Y",
"yQQQQ": "QQQQ Y",
"yw-count-one": "שבוע w בשנת Y",
"yw-count-two": "שבוע w בשנת Y",
"yw-count-many": "שבוע w בשנת Y",
"yw-count-other": "שבוע w בשנת Y"
},
"appendItems": {
  "Day": "{0} ({2}: {1})",
  "Day-Of-Week": "{0} {1}",
  "Era": "{1} {0}",
  "Hour": "{0} ({2}: {1})",
  "Minute": "{0} ({2}: {1})",
  "Month": "{0} ({2}: {1})",
  "Quarter": "{0} ({2}: {1})",
  "Second": "{0} ({2}: {1})",
  "Timezone": "{0} {1}",
  "Week": "{0} ({2}: {1})",
  "Year": "{1} {0}"
},
"intervalFormats": {
  "intervalFormatFallback": "{0} - {1}",
  "d": {
    "d": "d-d"
  },
  "h": {
    "a": "h a - h a",
    "h": "h-h a"
  },
  "H": {
    "H": "H-H"
  },
  "hm": {
    "a": "h:mm a - h:mm a",
    "h": "h:mm-h:mm a",
    "m": "h:mm-h:mm a"
  },
  "Hm": {

```

```

        "H": "H:mm-H:mm",
        "m": "H:mm-H:mm"
    },
    "hmv": {
        "a": "h:mm a - h:mm a v",
        "h": "h:mm-h:mm a v",
        "m": "h:mm-h:mm a v"
    },
    "Hmv": {
        "H": "H:mm-H:mm v",
        "m": "H:mm-H:mm v"
    },
    "hv": {
        "a": "h a - h a v",
        "h": "h-h a v"
    },
    "Hv": {
        "H": "H-H v"
    },
    "M": {
        "M": "M-M"
    },
    "Md": {
        "d": "d.M-d.M",
        "M": "d.M-d.M"
    },
    "MEd": {
        "d": "EEEE d.M-EEEE d.M",
        "M": "EEEE d.M - EEEE d.M"
    },
    "MMM": {
        "M": "MMM-MMM"
    },
    "MMMd": {
        "d": "d-d 1MMM",
        "M": "d 1MMM-d 1MMM"
    },
    "MMMEd": {
        "d": "EEEE, d 1MMM - EEEE, d 1MMM",
        "M": "EEEE, d 1MMM - EEEE, d 1MMM"
    },
    "MMMM": {
        "M": "LLLL-LLLL"
    },
    "Y": {
        "Y": "Y-Y"
    },
    "YM": {
        "M": "M.Y-M.Y",
        "Y": "M.Y-M.Y"
    },
    "YMd": {
        "d": "dd.M.y - dd.M.y",
        "M": "d.M.y - d.M.y",
        "Y": "d.M.y - d.M.y"
    },
    "yMEd": {

```

```
"d": "EEEE d.M.y - EEEE d.M.y",
"M": "EEEE d.M.y - EEEE d.M.y",
"y": "EEEE d.M.y - EEEE d.M.y"
},
"yMMM": {
    "M": "MMM-MMM y",
    "y": "MMM y - MMM y"
},
"yMMMM": {
    "d": "d-d MMMM y",
    "M": "d MMM - d MMM y",
    "y": "d MMM y - d MMM y"
},
"yMMMd": {
    "d": "EEEE d MMM - EEEE d MMM y",
    "M": "EEEE d MMM - EEEE d MMM y",
    "y": "EEEE d MMM y - EEEE d MMM y"
},
"yMMMM": {
    "M": "MMMM-MMMM y",
    "y": "MMMM y-MMMM y"
}
}
```

CA-GREGORIAN.JSX

```
{
    "main";
    {
        "he";
        {
            "identity";
            {
                "version";
                {
                    "_number";
                    "$Revision: 13259 $",
                    "_cldrVersion";
                    "31.0.1";
                }
                "language";
                "he";
            }
        }
        "dates";
        {
            "calendars";
            {
                "gregorian";
                {
```

```

    "months";
    {
      "format";
      {
        "abbreviated";
        {
          "1";
          "ינו'",
          "2";
          "פבר'",
          "3";
          "מרץ",
          "4";
          "אפר'",
          "5";
          "מאי",
          "6";
          "יוני",
          "7";
          "יולי",
          "8";
          "אוג'",
          "9";
          "ספט'",
          "10";
          "אוק'",
          "11";
          "נוב'",
          "12";
          "דצמ'";
        }
        "narrow";
        {
          "1";
          "1",
          "2";
          "2",
          "3";
          "3",
          "4";
          "4",
          "5";
          "5",
          "6";
          "6",
          "7";
          "7",
          "8";
          "8",
          "9";
          "9",
          "10";
          "10",
          "11";
          "11",
          "12";
          "12";
        }
      }
    }
  
```

```

    }
    "wide";
    {
        "1";
        "ינואר",
        "2";
        "פברואר",
        "3";
        "מרץ",
        "4";
        "אפריל",
        "5";
        "מאי",
        "6";
        "יוני",
        "7";
        "יולי",
        "8";
        "אוגוסט",
        "9";
        "ספטמבר",
        "10";
        "אוקטובר",
        "11";
        "נובמבר",
        "12";
        "דצמבר";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "1";
        "ינו'",
        "2";
        "פבר'",
        "3";
        "מרץ",
        "4";
        "אפר'",
        "5";
        "מאי",
        "6";
        "יוני",
        "7";
        "יולי",
        "8";
        "אוג'",
        "9";
        "ספט'",
        "10";
        "אוק'",
        "11";
        "נוב'",
        "12";
        "דצמ'";
    }
}

```

```
}  
"narrow";  
{  
  "1";  
  "1",  
    "2";  
  "2",  
    "3";  
  "3",  
    "4";  
  "4",  
    "5";  
  "5",  
    "6";  
  "6",  
    "7";  
  "7",  
    "8";  
  "8",  
    "9";  
  "9",  
    "10";  
  "10",  
    "11";  
  "11",  
    "12";  
  "12";  
}  
"wide";  
{  
  "1";  
  "ינואר",  
    "2";  
  "פברואר",  
    "3";  
  "מרץ",  
    "4";  
  "אפריל",  
    "5";  
  "מאי",  
    "6";  
  "יוני",  
    "7";  
  "יולי",  
    "8";  
  "אוגוסט",  
    "9";  
  "ספטמבר",  
    "10";  
  "אוקטובר",  
    "11";  
  "נובמבר",  
    "12";  
  "דצמבר";  
}  
}
```

```
"days";
{
  "format";
  {
    "abbreviated";
    {
      "sun";
      "א' יום",
      "mon";
      "ב' יום",
      "tue";
      "ג' יום",
      "wed";
      "ד' יום",
      "thu";
      "ה' יום",
      "fri";
      "ו' יום",
      "sat";
      "שבת";
    }
    "narrow";
    {
      "sun";
      "א'",
      "mon";
      "ב'",
      "tue";
      "ג'",
      "wed";
      "ד'",
      "thu";
      "ה'",
      "fri";
      "ו'",
      "sat";
      "ש";
    }
    "short";
    {
      "sun";
      "א'",
      "mon";
      "ב'",
      "tue";
      "ג'",
      "wed";
      "ד'",
      "thu";
      "ה'",
      "fri";
      "ו'",
      "sat";
      "ש";
    }
  }
  "wide";
  {
```

```

        "sun";
        "יום ראשון",
        "mon";
        "יום שני",
        "tue";
        "יום שלישי",
        "wed";
        "יום רביעי",
        "thu";
        "יום חמישי",
        "fri";
        "יום שישי",
        "sat";
        "יום שבת";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "sun";
        "יום א'",
        "mon";
        "יום ב'",
        "tue";
        "יום ג'",
        "wed";
        "יום ד'",
        "thu";
        "יום ה'",
        "fri";
        "יום ו'",
        "sat";
        "שבת";
    }
}
"narrow";
{
    "sun";
    "א'",
    "mon";
    "ב'",
    "tue";
    "ג'",
    "wed";
    "ד'",
    "thu";
    "ה'",
    "fri";
    "ו'",
    "sat";
    "ש";
}
"short";
{
    "sun";
    "א'",
    "mon";

```



```

        "ב'",
        "tue";
        "ג'",
        "wed";
        "ד'",
        "thu";
        "ה'",
        "fri";
        "ו'",
        "sat";
        "ש'";
    }
    "wide";
    {
        "sun";
        "יום ראשון",
        "mon";
        "יום שני",
        "tue";
        "יום שלישי",
        "wed";
        "יום רביעי",
        "thu";
        "יום חמישי",
        "fri";
        "יום שישי",
        "sat";
        "יום שבת";
    }
}
"quarters";
{
    "format";
    {
        "abbreviated";
        {
            "1";
            "Q1",
            "2";
            "Q2",
            "3";
            "Q3",
            "4";
            "Q4";
        }
        "narrow";
        {
            "1";
            "1",
            "2";
            "2",
            "3";
            "3",
            "4";
            "4";
        }
    }
}

```

```

        "wide";
        {
            "1";
            "1 רבעון",
            "2";
            "2 רבעון",
            "3";
            "3 רבעון",
            "4";
            "4 רבעון";
        }
    }
    "stand-alone";
    {
        "abbreviated";
        {
            "1";
            "Q1",
            "2";
            "Q2",
            "3";
            "Q3",
            "4";
            "Q4";
        }
        "narrow";
        {
            "1";
            "1",
            "2";
            "2",
            "3";
            "3",
            "4";
            "4";
        }
        "wide";
        {
            "1";
            "1 רבעון",
            "2";
            "2 רבעון",
            "3";
            "3 רבעון",
            "4";
            "4 רבעון";
        }
    }
}
"dayPeriods";
{
    "format";
    {
        "abbreviated";
        {
            "midnight";
            "מצות",

```

```

        "am";
        "לפנה"צ",
        "pm";
        "אחה"צ",
        "morning1";
        "בוקר",
        "afternoon1";
        "צהריים",
        "afternoon2";
        "אחר הצהריים",
        "evening1";
        "ערב",
        "night1";
        "לילה",
        "night2";
        "לפנות בוקר";
    }
    "narrow";
    {
        "midnight";
        "חצות",
        "am";
        "לפנה"צ",
        "pm";
        "אחה"צ",
        "morning1";
        "בוקר",
        "afternoon1";
        "צהריים",
        "afternoon2";
        "אחר הצהריים",
        "evening1";
        "ערב",
        "night1";
        "לילה",
        "night2";
        "לפנות בוקר";
    }
    "wide";
    {
        "midnight";
        "חצות",
        "am";
        "לפנה"צ",
        "pm";
        "אחה"צ",
        "morning1";
        "בוקר",
        "afternoon1";
        "צהריים",
        "afternoon2";
        "אחר הצהריים",
        "evening1";
        "ערב",
        "night1";
        "לילה",
        "night2";
    }

```

```

        "לפנות בוקר";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "midnight";
        "חצות",
        "am";
        "לפנה"צ",
        "pm";
        "אחה"צ",
        "morning1";
        "בוקר",
        "afternoon1";
        "צהריים",
        "afternoon2";
        "אחר הצהריים",
        "evening1";
        "ערב",
        "night1";
        "לילה",
        "night2";
        "לפנות בוקר";
    }
}
"narrow";
{
    "midnight";
    "חצות",
    "am";
    "לפנה"צ",
    "pm";
    "אחה"צ",
    "morning1";
    "בוקר",
    "afternoon1";
    "צהריים",
    "afternoon2";
    "אחר הצהריים",
    "evening1";
    "ערב",
    "night1";
    "לילה",
    "night2";
    "לפנות בוקר";
}
"wide";
{
    "midnight";
    "חצות",
    "am";
    "לפנה"צ",
    "pm";
    "אחה"צ",
    "morning1";
    "בוקר",

```

```

        "afternoon1";
        "צהריים",
        "afternoon2";
        "אחר הצהריים",
        "evening1";
        "ערב",
        "night1";
        "לילה",
        "night2";
        "לפנות בוקר";
    }
}
"eras";
{
    "eraNames";
    {
        "0";
        "לפני הספירה",
        "0-alt-variant";
        "לפנה"ס",
        "1";
        "לספירה",
        "1-alt-variant";
        "CE";
    }
    "eraAbbr";
    {
        "0";
        "לפנה"ס",
        "0-alt-variant";
        "BCE",
        "1";
        "לספירה",
        "1-alt-variant";
        "CE";
    }
    "eraNarrow";
    {
        "0";
        "לפנה"ס",
        "0-alt-variant";
        "BCE",
        "1";
        "לספירה",
        "1-alt-variant";
        "CE";
    }
}
"dateFormats";
{
    "full";
    "EEEE, d MMMM y",
    "long";
    "d MMMM y",
    "medium";
    "d MMM y",

```

```

        "short";
        "d.M.y";
    }
    "timeFormats";
    {
        "full";
        "H:mm:ss zzzz",
        "long";
        "H:mm:ss z",
        "medium";
        "H:mm:ss",
        "short";
        "H:mm";
    }
    "dateTimeFormats";
    {
        "full";
        "{1} בשעה {0}",
        "long";
        "{1} בשעה {0}",
        "medium";
        "{1}, {0}",
        "short";
        "{1}, {0}",
        "availableFormats";
        {
            "d";
            "d",
            "E";
            "ccc",
            "Ed";
            "E ה-d",
            "Ehm";
            "E h:mm a",
            "EHm";
            "E H:mm",
            "Ehms";
            "E h:mm:ss a",
            "EHms";
            "E H:mm:ss",
            "Gy";
            "y G",
            "GyMMM";
            "MMM y G",
            "GyMMMd";
            "d הMMM y G",
            "GyMMMED";
            "E, d הMMM y G",
            "h";
            "h a",
            "H";
            "H",
            "hm";
            "h:mm a",
            "Hm";
            "H:mm",
            "hms";
        }
    }

```

```

    "h:mm:ss a",
        "Hms";
    "H:mm:ss",
        "hmsv";
    "h:mm:ss a v",
        "Hmsv";
    "HH:mm:ss v",
        "hmv";
    "h:mm a v",
        "Hmv";
    "HH:mm v",
        "M";
    "L",
        "Md";
    "d.M",
        "MEd";
    "E, d.M",
        "MMM";
    "LLL",
        "MMMd";
    "d ׁMMM",
        "MMMEd";
    "E, d ׁMMM",
        "MMMMd";
    "d ׁMMMM",
        "MMMMW-count-one";
    "שבוט W ׁMMM",
        "MMMMW-count-two";
    "שבוט W ׁMMM",
        "MMMMW-count-many";
    "שבוט W ׁMMM",
        "MMMMW-count-other";
    "שבוט W ׁMMM",
        "ms";
    "mm:ss",
        "y";
    "y",
        "yM";
    "M.y",
        "yMd";
    "d.M.y",
        "yMEd";
    "E, d.M.y",
        "yMM";
    "M.y",
        "yMMM";
    "MMM y",
        "yMMMd";
    "d ׁMMM y",
        "yMMMEd";
    "E, d ׁMMM y",
        "yMMMM";
    "MMMM y",
        "yQQQ";
    "QQQ y",
        "yQQQQ";
    "QQQQ y",

```

```

        "yw-count-one";
        "yw-count-two";
        "yw-count-many";
        "yw-count-other";
        "yw-count-other";
    }
    "appendItems";
    {
        "Day";
        "{0} ({2}: {1})",
        "Day-Of-Week";
        "{0} {1}",
        "Era";
        "{1} {0}",
        "Hour";
        "{0} ({2}: {1})",
        "Minute";
        "{0} ({2}: {1})",
        "Month";
        "{0} ({2}: {1})",
        "Quarter";
        "{0} ({2}: {1})",
        "Second";
        "{0} ({2}: {1})",
        "Timezone";
        "{0} {1}",
        "Week";
        "{0} ({2}: {1})",
        "Year";
        "{1} {0}";
    }
    "intervalFormats";
    {
        "intervalFormatFallback";
        "{0} - {1}",
        "d";
        {
            "d";
            "d-d";
        }
        "h";
        {
            "a";
            "h a - h a",
            "h";
            "h-h a";
        }
        "H";
        {
            "H";
            "H-H";
        }
        "hm";
        {

```



```

        "a";
        "h:mm a - h:mm a",
        "h";
        "h:mm-h:mm a",
        "m";
        "h:mm-h:mm a";
    }
    "Hm";
    {
        "H";
        "H:mm-H:mm",
        "m";
        "H:mm-H:mm";
    }
    "hmv";
    {
        "a";
        "h:mm a - h:mm a v",
        "h";
        "h:mm-h:mm a v",
        "m";
        "h:mm-h:mm a v";
    }
    "Hmv";
    {
        "H";
        "H:mm-H:mm v",
        "m";
        "H:mm-H:mm v";
    }
    "hv";
    {
        "a";
        "h a - h a v",
        "h";
        "h-h a v";
    }
    "Hv";
    {
        "H";
        "H-H v";
    }
    "M";
    {
        "M";
        "M-M";
    }
    "Md";
    {
        "d";
        "d.M-d.M",
        "M";
        "d.M-d.M";
    }
    "MEd";
    {
        "d";

```

```

        "EEEE d.M-EEEE d.M",
        "M";
        "EEEE d.M - EEEE d.M";
    }
    "MMM";
    {
        "M";
        "MMM-MMM";
    }
    "MMMd";
    {
        "d";
        "d-d ׀MMM",
        "M";
        "d ׀MMM-d ׀MMM";
    }
    "MMMEd";
    {
        "d";
        "EEEE, d ׀MMM - EEEE, d ׀MMM",
        "M";
        "EEEE, d ׀MMM - EEEE, d ׀MMM";
    }
    "MMMM";
    {
        "M";
        "LLLL-LLLL";
    }
    "Y";
    {
        "Y";
        "Y-Y";
    }
    "yM";
    {
        "M";
        "M.y-M.y",
        "Y";
        "M.y-M.y";
    }
    "yMd";
    {
        "d";
        "dd.M.y - dd.M.y",
        "M";
        "d.M.y - d.M.y",
        "Y";
        "d.M.y - d.M.y";
    }
    "yMEd";
    {
        "d";
        "EEEE d.M.y - EEEE d.M.y",
        "M";
        "EEEE d.M.y - EEEE d.M.y",
        "Y";
        "EEEE d.M.y - EEEE d.M.y";
    }

```

```

}
"yMMM";
{
    "M";
    "MMM-MMM y",
        "Y";
    "MMM y - MMM Y";
}
"yMMMd";
{
    "d";
    "d-d ∩MMM y",
        "M";
    "d MMM - d MMM y",
        "Y";
    "d MMM y - d MMM Y";
}
"yMMMEd";
{
    "d";
    "EEEE d MMM - EEEE d MMM y",
        "M";
    "EEEE d MMM - EEEE d MMM y",
        "Y";
    "EEEE d MMM y - EEEE d MMM Y";
}
"yMMMM";
{
    "M";
    "MMMM-MMMM y",
        "Y";
    "MMMM y-MMMM Y";
}
}
}
}
}
}
}
}
}
}
}

```

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
//Load the L10n, loadCldr from ej2-base
import { loadCldr, L10n } from '@syncfusion/ej2-base';
import { DatePickerComponent } from '@syncfusion/ej2-react-calendars';
//load the CLDR data files.
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as heTimeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, heTimeZoneNames);
```

```

L10n.load({
  'he': {
    'daterangepicker': {
      applyText: 'להחיל טקסט',
      cancelText: 'בטל טקסט',
      customRange: 'טווח מותאם אישית',
      days: 'ימים',
      endLabel: 'ח',
      placeholder: 'בחר טווח',
      selectedDays: 'ימים נבחרים',
      startLabel: 'תווית התחלה'
    }
  }
});
//import the datangepicker component
class App extends React.Component {
  render() {
    return <DateRangePickerComponent id="daterangepicker" locale='he'
enableRtl={true}/>;
  }
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
//Load the L10n, loadCldr from ej2-base
import { loadCldr, L10n } from '@syncfusion/ej2-base';
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
//load the CLDR data files.
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as heTimeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, heTimeZoneNames);
L10n.load({
  'he': {
    'daterangepicker': {
      applyText: 'להחיל טקסט',
      cancelText: 'בטל טקסט',
      customRange: 'טווח מותאם אישית',
      days: 'ימים',
      endLabel: 'ח',
      placeholder: 'בחר טווח',
      selectedDays: 'ימים נבחרים',
      startLabel: 'תווית התחלה'
    }
  }
});
//import the datangepicker component
class App extends React.Component<{}, {}> {
  render() {

```

```

    return <DatePickerComponent id="daterangepicker" locale='he'
enableRtl={true} />
  }
};
ReactDOM.render(<App />, document.getElementById('element'));

```

NUMBERINGSYSTEMS.JSON

```
{
  "supplemental": {
    "version": {
      "_number": "$Revision: 12732 $",
      "_unicodeVersion": "9.0.0",
      "_cldrVersion": "31"
    },
    "numberingSystems": {
      "adlm": {
        "_digits": "ᐃᐅᐆᐇᐈᑦᑉᑊᑋᑌ",
        "_type": "numeric"
      },
      "ahom": {
        "_digits": "ᩁᩂᩃᩄᩅᩆᩇᩈᩉᩊᩋᩌᩍᩎᩏᩐᩑᩒᩓᩔᩕᩖᩗᩘᩙᩚᩛᩜᩝᩞ᩟᩠ᩡᩢᩣᩤᩥᩦᩧᩨᩩᩪᩫᩬᩭᩮᩯᩰᩱᩲᩳᩴ᩵᩶᩷᩸᩹᩺᩻᩼᩽᩾᩿",
        "_type": "numeric"
      },
      "arab": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "arabext": {
        "_digits": "ا ب ج د ه و ز ح ط ق ك ل م ن ي",
        "_type": "numeric"
      },
      "armn": {
        "_rules": "armenian-upper",
        "_type": "algorithmic"
      },
      "armnlow": {
        "_rules": "armenian-lower",
        "_type": "algorithmic"
      },
      "bali": {
        "_digits": "᭐ᭀᭁᭂᭃ᭄ᭅᭆᭇᭈᭉᭊᭋᭌ᭍᭎᭏᭐᭑᭒᭓᭔᭕᭖᭗᭘᭙᭚᭛᭜᭝᭞᭟᭠᭡᭢᭣᭤᭥᭦᭧᭨᭩᭪᭬᭫᭭᭮᭯᭰᭱᭲᭳᭴᭵᭶᭷᭸᭹᭺᭻᭼᭽᭾᭿",
        "_type": "numeric"
      },
      "beng": {
        "_digits": "০১২৩৪৫৬৭৮৯",
        "_type": "numeric"
      },
      "bhks": {
        "_digits": "ႤႬႭᆀᆁᆂᆃᆄᆅᆆᆇᆈᆉᆊᆋᆌᆍᆎᆏᆐᆑᆒᆓᆔᆕᆖᆗᆘᆙᆚᆛᆜᆝᆞᆟᆠᆡᆢᆣᆤᆥᆦᆧᆨᆩᆪᆫᆬᆭᆮᆯᆰᆱᆲᆳᆴᆵᆶᆷᆸᆹᆺᆻᆼᆽᆾᆿ",
        "_type": "numeric"
      },
      "brah": {
        "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍᱎᱏ᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱹᱺᱻᱼᱽ᱾᱿",
        "_type": "numeric"
      }
    }
  }
}
```

```
"cakh": {
    "_digits": "Ⴍᄂᅆᅉᅈᅇᅋᅌ",
    "_type": "numeric"
},
"cham": {
    "_digits": "ꠘꠙꠚꠛꠜꠝꠞꠟ",
    "_type": "numeric"
},
"cyril": {
    "_rules": "cyrillic-lower",
    "_type": "algorithmic"
},
"deva": {
    "_digits": "ॐ॒॑॓॔ॕॖॗक़",
    "_type": "numeric"
},
"ethi": {
    "_rules": "ethiopic",
    "_type": "algorithmic"
},
"fullwide": {
    "_digits": "0 1 2 3 4 5 6 7 8 9 ",
    "_type": "numeric"
},
"geor": {
    "_rules": "georgian",
    "_type": "algorithmic"
},
"grek": {
    "_rules": "greek-upper",
    "_type": "algorithmic"
},
"greklow": {
    "_rules": "greek-lower",
    "_type": "algorithmic"
},
"gujr": {
    "_digits": "૦૧૨૩૪૫૬૭૮૯",
    "_type": "numeric"
},
"guru": {
    "_digits": "੦੧੨੩੪੫੬੭੮੯",
    "_type": "numeric"
},
"hanidays": {
    "_rules": "zh/SpelloutRules/spellout-numbering-days",
    "_type": "algorithmic"
},
"hanidec": {
    "_digits": "〇一二三四五六七八九",
    "_type": "numeric"
},
"hans": {
    "_rules": "zh/SpelloutRules/spellout-cardinal",
```

```

    "_type": "algorithmic"
  },
  "hansfin": {
    "_rules": "zh/SpelloutRules/spellout-cardinal-financial",
    "_type": "algorithmic"
  },
  "hant": {
    "_rules": "zh_Hant/SpelloutRules/spellout-cardinal",
    "_type": "algorithmic"
  },
  "hantfin": {
    "_rules": "zh_Hant/SpelloutRules/spellout-cardinal-financial",
    "_type": "algorithmic"
  },
  "hebr": {
    "_rules": "hebrew",
    "_type": "algorithmic"
  },
  "hmng": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "java": {
    "_digits": "௦௧௨௩௪௫௬௭௮௯௦௧௨௩௪௫௬௭௮௯௦",
    "_type": "numeric"
  },
  "jpan": {
    "_rules": "ja/SpelloutRules/spellout-cardinal",
    "_type": "algorithmic"
  },
  "jpanfin": {
    "_rules": "ja/SpelloutRules/spellout-cardinal-financial",
    "_type": "algorithmic"
  },
  "kali": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "khmr": {
    "_digits": "០១២៣៤៥៦៧៨៩",
    "_type": "numeric"
  },
  "knda": {
    "_digits": "೦೧೨೩೪೫೬೭೮೯",
    "_type": "numeric"
  },
  "lana": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "lanatham": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "lao": {

```

```

    "_digits": "൦൧൨൩൪൫൬൭൮൯",
    "_type": "numeric"
  },
  "latn": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "lepc": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "limb": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "mathbold": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mathdbl": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mathmono": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mathsanb": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mathsans": {
    "_digits": "0123456789",
    "_type": "numeric"
  },
  "mlym": {
    "_digits": "൦൧൨൩൪൫൬൭൮൯",
    "_type": "numeric"
  },
  "modi": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "mong": {
    "_digits": "᠐᠑᠒᠓᠐ᠠᠨᠨᠠᠭ",
    "_type": "numeric"
  },
  "mroo": {
    "_digits": "□□□□□□□□□□",
    "_type": "numeric"
  },
  "mtei": {
    "_digits": "ႤႤႤႤႤႤႤႤႤႤ",
    "_type": "numeric"
  },

```



```

    "mymr": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "mymrshan": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "mymrtlng": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "newa": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "nkoo": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "olck": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "orya": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "osma": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "roman": {
      "_rules": "roman-upper",
      "_type": "algorithmic"
    },
    "romanlow": {
      "_rules": "roman-lower",
      "_type": "algorithmic"
    },
    "saur": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "shrd": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "sind": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",
      "_type": "numeric"
    },
    "sinh": {
      "_digits": "၀၁၂၃၄၅၆၇၈၉",

```

```

    "_type": "numeric"
  },
  "sora": {
    "_digits": "ᱵᱚᱠᱟᱨᱢᱟ",
    "_type": "numeric"
  },
  "sund": {
    "_digits": "ᱵᱚᱠᱟᱨᱢᱟ",
    "_type": "numeric"
  },
  "takr": {
    "_digits": "ᱵᱚᱠᱟᱨᱢᱟ",
    "_type": "numeric"
  },
  "talv": {
    "_digits": "ᱵᱚᱠᱟᱨᱢᱟ",
    "_type": "numeric"
  },
  "taml": {
    "_rules": "tamil",
    "_type": "algorithmic"
  },
  "tamldec": {
    "_digits": "0௧௨௩௪௫௬௭௮௯",
    "_type": "numeric"
  },
  "telu": {
    "_digits": "౦౧౨౩౪౫౬౭౮౯",
    "_type": "numeric"
  },
  "thai": {
    "_digits": "๐๑๒๓๔๕๖๗๘๙",
    "_type": "numeric"
  },
  "tibb": {
    "_digits": "༠༡༢༣༤༥༦༧༨༩",
    "_type": "numeric"
  },
  "tirh": {
    "_digits": "ᱵᱚᱠᱟᱨᱢᱟ",
    "_type": "numeric"
  },
  "vair": {
    "_digits": "ᱵᱚᱠᱟᱨᱢᱟ",
    "_type": "numeric"
  },
  "wara": {
    "_digits": "ᱵᱚᱠᱟᱨᱢᱟ",
    "_type": "numeric"
  }
}

```

NUMBERINGSYSTEMS.JSX

```

{
  "supplemental";
  {
    "version";
    {
      "_number";
      "$Revision: 12732 $",
      "_unicodeVersion";
      "9.0.0",
      "_cldrVersion";
      "31";
    }
    "numberingSystems";
    {
      "adlm";
      {
        "_digits";
        "ᲐᲑᲒᲓᲔᲕᲖᲗᲘᲙ",
        "_type";
        "numeric";
      }
      "ahom";
      {
        "_digits";
        "ᩌᩍᩎᩏᩐᩑᩒᩓᩔᩕᩖᩗᩘᩙᩚᩛᩜᩝᩞ᩟᩠ᩡᩢᩣᩤᩥᩦᩧᩨᩩᩪᩫᩬᩭᩮᩯᩰᩱᩲᩳᩴ᩵᩶᩷᩸᩹᩺᩻᩼᩽᩾᩿",
        "_type";
        "numeric";
      }
      "arab";
      {
        "_digits";
        "٠١٢٣٤٥٦٧٨٩",
        "_type";
        "numeric";
      }
      "arabext";
      {
        "_digits";
        "٠١٢٣٤٥٦٧٨٩",
        "_type";
        "numeric";
      }
      "armn";
      {
        "_rules";
        "armenian-upper",
        "_type";
        "algorithmic";
      }
      "armnlow";
      {
        "_rules";
        "armenian-lower",
        "_type";
        "algorithmic";
      }
    }
  }
}

```

```

    }
    "bali";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "beng";
    {
        "_digits";
        "০১২৩৪৫৬৭৮৯",
        "_type";
        "numeric";
    }
    "bhks";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "brah";
    {
        "_digits";
        "·ᱠᱡᱢᱫᱷᱚᱴᱚᱨ",
        "_type";
        "numeric";
    }
    "cakm";
    {
        "_digits";
        "ႤႬႬႬႬႬႬႬႬႬ",
        "_type";
        "numeric";
    }
    "cham";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "cyr1";
    {
        "_rules";
        "cyrillic-lower",
        "_type";
        "algorithmic";
    }
    "deva";
    {
        "_digits";
        "०१२३४५६७८९",
        "_type";
        "numeric";
    }

```

```
}
"ethi";
{
  "_rules";
  "ethiopic",
  "_type";
  "algorithmic";
}
"fullwide";
{
  "_digits";
  "0 1 2 3 4 5 6 7 8 9",
  "_type";
  "numeric";
}
"geor";
{
  "_rules";
  "georgian",
  "_type";
  "algorithmic";
}
"grek";
{
  "_rules";
  "greek-upper",
  "_type";
  "algorithmic";
}
"greklow";
{
  "_rules";
  "greek-lower",
  "_type";
  "algorithmic";
}
"gujr";
{
  "_digits";
  "૦૧૨૩૪૫૬૭૮૯",
  "_type";
  "numeric";
}
"guru";
{
  "_digits";
  "੦੧੨੩੪੫੬੭੮੯",
  "_type";
  "numeric";
}
"hanidays";
{
  "_rules";
  "zh/SpelloutRules/spellout-numbering-days",
  "_type";
  "algorithmic";
}
```

```
"hanidec";
{
    "_digits";
    "〇一二三四五六七八九",
        "_type";
    "numeric";
}
"hans";
{
    "_rules";
    "zh/SpelloutRules/spellout-cardinal",
        "_type";
    "algorithmic";
}
"hansfin";
{
    "_rules";
    "zh/SpelloutRules/spellout-cardinal-financial",
        "_type";
    "algorithmic";
}
"hant";
{
    "_rules";
    "zh_Hant/SpelloutRules/spellout-cardinal",
        "_type";
    "algorithmic";
}
"hantfin";
{
    "_rules";
    "zh_Hant/SpelloutRules/spellout-cardinal-financial",
        "_type";
    "algorithmic";
}
"hebr";
{
    "_rules";
    "hebrew",
        "_type";
    "algorithmic";
}
"hmng";
{
    "_digits";
    "᠐᠑᠒᠓᠔᠕᠖᠗᠘᠙",
        "_type";
    "numeric";
}
"java";
{
    "_digits";
    "᠐ᠠᠨᠵᠢᠰᠤᠷᠦᠭᠡᠯᠦᠳᠡᠮᠤᠩᠭᠡ",
        " type";
```

```

        "numeric";
    }
    "jpan";
    {
        "_rules";
        "ja/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "jpanfin";
    {
        "_rules";
        "ja/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "kali";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "khrm";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈၉",
        "_type";
        "numeric";
    }
    "knda";
    {
        "_digits";
        "೦೧೨೩೪೫೬೭೮೯",
        "_type";
        "numeric";
    }
    "lana";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "lanatham";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "lao";
    {
        "_digits";
        "໐໑໒໓໔໕໖໗໘໙",
        "_type";

```

```

        "numeric";
    }
    "latn";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "lepc";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
    "limb";
    {
        "_digits";
        "□□□□□□□□□□",
        "_type";
        "numeric";
    }
    "mathbold";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathdbl";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathmono";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathsanb";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathsans";
    {
        "_digits";
        "0123456789",
        "_type";

```



```
"numeric";
}
"mlym";
{
    "_digits";
    "൦൧൨൩൪൫൬൭൮൯",
    "_type";
    "numeric";
}
"modi";
{
    "_digits";
    "□□□□□□□□□□",
    "_type";
    "numeric";
}
"mong";
{
    "_digits";
    "᠐᠑᠒᠓᠔᠕᠖᠗᠘᠙",
    "_type";
    "numeric";
}
"mroo";
{
    "_digits";
    "□□□□□□□□□□",
    "_type";
    "numeric";
}
"mtei";
{
    "_digits";
    "ႤႬႭᆞᇀᇁᇂᇃᇄᇅᇆᇇᇈᇉᇊᇋᇌᇍᇎᇏᇐᇑᇒᇓᇔᇕᇖᇗᇘᇙᇚᇛᇜᇝᇞᇟᇠᇡᇢᇣᇤᇥᇦᇧᇨᇩᇪᇫᇬᇭᇮᇯᇰᇱᇲᇳᇴᇶᇷᇸᇹᇺᇻᇼᇽᇾᇿ",
    "_type";
    "numeric";
}
"mymr";
{
    "_digits";
    "၀၁၂၃၄၅၆၇၈၉",
    "_type";
    "numeric";
}
"mymrshan";
{
    "_digits";
    "0123456789",
    "_type";
    "numeric";
}
"mymrtlng";
{
    " digits";
```

[illegible]

```

        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "shrd";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "sind";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "sinh";
    {
        "_digits";
        "⁂௯௯௯௮௮௮௮௮",
        "_type";
        "numeric";
    }
    "sora";
    {
        "_digits";
        "0௭௭௭௭௭௭௭",
        "_type";
        "numeric";
    }
    "sund";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "takr";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "talv";
    {
        "_digits";
        "⁂௮௮௮௮௮௮௮",
        "_type";
        "numeric";
    }
    "taml";
    {

```

```

        "_rules";
        "tamil",
        "_type";
        "algorithmic";
    }
    "tamldec";
    {
        "_digits";
        "0௧௨௩௪௫௬௭௮௯",
        "_type";
        "numeric";
    }
    "telu";
    {
        "_digits";
        "౦౧౨౩౪౫౬౭౮౯",
        "_type";
        "numeric";
    }
    "thai";
    {
        "_digits";
        "๐๑๒๓๔๕๖๗๘๙",
        "_type";
        "numeric";
    }
    "tibtb";
    {
        "_digits";
        "༠༡༢༣༤༥༦༧༨༩",
        "_type";
        "numeric";
    }
    "tirh";
    {
        "_digits";
        "୦୧୨୩୪୫୬୭୮୯",
        "_type";
        "numeric";
    }
    "vaih";
    {
        "_digits";
        "୦୧୨୩୪୫୬୭୮୯",
        "_type";
        "numeric";
    }
    "wara";
    {
        "_digits";
        "୦୧୨୩୪୫୬୭୮୯",
        "_type";
        "numeric";
    }
    }
}

```

```
}
```

NUMBERS.JSON

```
{
  "main": {
    "he": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31.0.1"
        },
        "language": "he"
      },
      "numbers": {
        "defaultNumberingSystem": "latn",
        "otherNumberingSystems": {
          "native": "latn",
          "traditional": "hebr"
        },
        "minimumGroupingDigits": "1",
        "symbols-numberSystem-latn": {
          "decimal": ".",
          "group": ",",
          "list": ";",
          "percentSign": "%",
          "plusSign": "+",
          "minusSign": "-",
          "exponential": "E",
          "superscriptingExponent": "×",
          "perMille": "‰",
          "infinity": "∞",
          "nan": "NaN",
          "timeSeparator": ":"
        },
        "decimalFormats-numberSystem-latn": {
          "standard": "#,##0.###",
          "long": {
            "decimalFormat": {
              "1000-count-one": "אלף 0",
              "1000-count-two": "אלף 0",
              "1000-count-many": "אלף 0",
              "1000-count-other": "אלף 0",
              "10000-count-one": "אלף 00",
              "10000-count-two": "אלף 00",
              "10000-count-many": "אלף 00",
              "10000-count-other": "אלף 00",
              "100000-count-one": "אלף 000",
              "100000-count-two": "אלף 000",
              "100000-count-many": "אלף 000",
              "100000-count-other": "אלף 000",
              "1000000-count-one": "מיליון 0",
              "1000000-count-two": "מיליון 0",
              "1000000-count-many": "מיליון 0",
              "1000000-count-other": "מיליון 0",
              "10000000-count-one": "מיליון 00",
              "10000000-count-two": "מיליון 00",
              "10000000-count-many": "מיליון 00",
              "10000000-count-other": "מיליון 00"
            }
          }
        }
      }
    }
  }
}
```

```

        "10000000-count-two": "00 מיליון",
        "10000000-count-many": "00 מיליון",
        "10000000-count-other": "00 מיליון",
        "100000000-count-one": "000 מיליון",
        "100000000-count-two": "000 מיליון",
        "100000000-count-many": "000 מיליון",
        "100000000-count-other": "000 מיליון",
        "1000000000-count-one": "0 מיליארד",
        "1000000000-count-two": "0 מיליארד",
        "1000000000-count-many": "0 מיליארד",
        "1000000000-count-other": "0 מיליארד",
        "10000000000-count-one": "00 מיליארד",
        "10000000000-count-two": "00 מיליארד",
        "10000000000-count-many": "00 מיליארד",
        "10000000000-count-other": "00 מיליארד",
        "100000000000-count-one": "000 מיליארד",
        "100000000000-count-two": "000 מיליארד",
        "100000000000-count-many": "000 מיליארד",
        "100000000000-count-other": "000 מיליארד",
        "1000000000000-count-one": "0 טריליון",
        "1000000000000-count-two": "0 טריליון",
        "1000000000000-count-many": "0 טריליון",
        "1000000000000-count-other": "0 טריליון",
        "10000000000000-count-one": "00 טריליון",
        "10000000000000-count-two": "00 טריליון",
        "10000000000000-count-many": "00 טריליון",
        "10000000000000-count-other": "00 טריליון",
        "100000000000000-count-one": "000 טריליון",
        "100000000000000-count-two": "000 טריליון",
        "100000000000000-count-many": "000 טריליון",
        "100000000000000-count-other": "000 טריליון"
    },
    },
    "short": {
        "decimalFormat": {
            "1000-count-one": "0K",
            "1000-count-two": "0K",
            "1000-count-many": "0K",
            "1000-count-other": "0K",
            "10000-count-one": "00K",
            "10000-count-two": "00K",
            "10000-count-many": "00K",
            "10000-count-other": "00K",
            "100000-count-one": "000K",
            "100000-count-two": "000K",
            "100000-count-many": "000K",
            "100000-count-other": "000K",
            "1000000-count-one": "0M",
            "1000000-count-two": "0M",
            "1000000-count-many": "0M",
            "1000000-count-other": "0M",
            "10000000-count-one": "00M",
            "10000000-count-two": "00M",
            "10000000-count-many": "00M",
            "10000000-count-other": "00M",
            "100000000-count-one": "000M",
            "100000000-count-two": "000M",

```

```

        "1000000000-count-many": "000M",
        "1000000000-count-other": "000M",
        "1000000000-count-one": "0B",
        "1000000000-count-two": "0B",
        "1000000000-count-many": "0B",
        "1000000000-count-other": "0B",
        "1000000000-count-one": "00B",
        "1000000000-count-two": "00B",
        "1000000000-count-many": "00B",
        "1000000000-count-other": "00B",
        "100000000000-count-one": "000B",
        "100000000000-count-two": "000B",
        "100000000000-count-many": "000B",
        "100000000000-count-other": "000B",
        "1000000000000-count-one": "0T",
        "1000000000000-count-two": "0T",
        "1000000000000-count-many": "0T",
        "1000000000000-count-other": "0T",
        "10000000000000-count-one": "00T",
        "10000000000000-count-two": "00T",
        "100000000000000-count-many": "00T",
        "100000000000000-count-other": "00T",
        "1000000000000000-count-one": "000T",
        "1000000000000000-count-two": "000T",
        "1000000000000000-count-many": "000T",
        "1000000000000000-count-other": "000T"
    }
}
},
"scientificFormats-numberSystem-latn": {
    "standard": "#E0"
},
"percentFormats-numberSystem-latn": {
    "standard": "#,##0%"
},
"currencyFormats-numberSystem-latn": {
    "currencySpacing": {
        "beforeCurrency": {
            "currencyMatch": "[:^S:]",
            "surroundingMatch": "[:digit:]",
            "insertBetween": " "
        },
        "afterCurrency": {
            "currencyMatch": "[:^S:]",
            "surroundingMatch": "[:digit:]",
            "insertBetween": " "
        }
    },
    "standard": "##0.00#, -¤; ##0.00#, ¤",
    "accounting": "#,##0.00 ¤",
    "short": {
        "standard": {
            "1000-count-one": "¤ 0K",
            "1000-count-two": "¤ 0K",
            "1000-count-many": "¤0K",
            "1000-count-other": "¤ 0K",
            "10000-count-one": "¤00K",

```

```

        "10000-count-two": "א00K",
        "10000-count-many": "א00K",
        "10000-count-other": "א 00K",
        "100000-count-one": "א000K",
        "100000-count-two": "א000K",
        "100000-count-many": "א000K",
        "100000-count-other": "א000K",
        "1000000-count-one": "א0M",
        "1000000-count-two": "א0M",
        "1000000-count-many": "א0M",
        "1000000-count-other": "א0M",
        "10000000-count-one": "א00M",
        "10000000-count-two": "א00M",
        "10000000-count-many": "א00M",
        "10000000-count-other": "א00M",
        "100000000-count-one": "א000M",
        "100000000-count-two": "א000M",
        "100000000-count-many": "א000M",
        "100000000-count-other": "א000M",
        "1000000000-count-one": "א0B",
        "1000000000-count-two": "א0B",
        "1000000000-count-many": "א0B",
        "1000000000-count-other": "א0B",
        "10000000000-count-one": "א00B",
        "10000000000-count-two": "א00B",
        "10000000000-count-many": "א00B",
        "10000000000-count-other": "א00B",
        "100000000000-count-one": "א000B",
        "100000000000-count-two": "א000B",
        "100000000000-count-many": "א000B",
        "100000000000-count-other": "א000B",
        "1000000000000-count-one": "א0T",
        "1000000000000-count-two": "א0T",
        "1000000000000-count-many": "א0T",
        "1000000000000-count-other": "א0T",
        "10000000000000-count-one": "א00T",
        "10000000000000-count-two": "א00T",
        "10000000000000-count-many": "א00T",
        "10000000000000-count-other": "א00T",
        "100000000000000-count-one": "א000T",
        "100000000000000-count-two": "א000T",
        "100000000000000-count-many": "א000T",
        "100000000000000-count-other": "א000T"
    },
    },
    "unitPattern-count-one": "{0} {1}",
    "unitPattern-count-two": "{0} {1}",
    "unitPattern-count-many": "{0} {1}",
    "unitPattern-count-other": "{0} {1}"
},
"miscPatterns-numberSystem-latn": {
    "atLeast": "≥{0}+",
    "range": "{0}-{1}"
},
"minimalPairs": {
    "pluralMinimalPairs": "שונה",
    "pluralMinimalPairs": "שנות ימים",

```



```

        "pluralMinimalPairs": "{0} שנה",
        "pluralMinimalPairs": "{0} שנים",
        "other": "פנה ימינה בפנייה ה-{0}"
    }
}
}
}
}
}

```

NUMBERS.JSX

```

{
    "main";
    {
        "he";
        {
            "identity";
            {
                "version";
                {
                    "_number";
                    "$Revision: 13259 $",
                    "_cldrVersion";
                    "31.0.1";
                }
                "language";
                "he";
            }
            "numbers";
            {
                "defaultNumberingSystem";
                "latn",
                "otherNumberingSystems";
                {
                    "native";
                    "latn",
                    "traditional";
                    "hebr";
                }
                "minimumGroupingDigits";
                "1",
                "symbols-numberSystem-latn";
                {
                    "decimal";
                    ".",
                    "group";
                    ",",
                    "list";
                    ";",
                    "percentSign";
                    "%",
                    "plusSign";
                    "+",
                    "minusSign";
                    "-",
                    "exponential";
                }
            }
        }
    }
}

```

```

        "E", "superscriptingExponent";
        "x", "perMille";
        "%", "infinity";
        "∞", "nan";
        "NaN", "timeSeparator";
        ":";
    }
    "decimalFormats-numberSystem-latn";
    {
        "standard";
        "#,##0.###",
        "long";
        {
            "decimalFormat";
            {
                "1000-count-one";
                "אלף 0",
                "1000-count-two";
                "אלף 0",
                "1000-count-many";
                "אלף 0",
                "1000-count-other";
                "אלף 0",
                "10000-count-one";
                "אלף 00",
                "10000-count-two";
                "אלף 00",
                "10000-count-many";
                "אלף 00",
                "10000-count-other";
                "אלף 00",
                "100000-count-one";
                "אלף 000",
                "100000-count-two";
                "אלף 000",
                "100000-count-many";
                "אלף 000",
                "100000-count-other";
                "אלף 000",
                "1000000-count-one";
                "מיליון 0",
                "1000000-count-two";
                "מיליון 0",
                "1000000-count-many";
                "מיליון 0",
                "1000000-count-other";
                "מיליון 0",
                "10000000-count-one";
                "מיליון 00",
                "10000000-count-two";
                "מיליון 00",
                "10000000-count-many";
            }
        }
    }

```

```
"00 מיליון",
    "10000000-count-other";
"00 מיליון",
    "100000000-count-one";
"000 מיליון",
    "1000000000-count-two";
"000 מיליון",
    "1000000000-count-many";
"000 מיליון",
    "1000000000-count-other";
"000 מיליון",
    "1000000000-count-one";
"0 מיליארד",
    "10000000000-count-two";
"0 מיליארד",
    "10000000000-count-many";
"0 מיליארד",
    "10000000000-count-other";
"0 מיליארד",
    "10000000000-count-one";
"00 מיליארד",
    "100000000000-count-two";
"00 מיליארד",
    "100000000000-count-many";
"00 מיליארד",
    "100000000000-count-other";
"00 מיליארד",
    "100000000000-count-one";
"00 מיליארד",
    "100000000000-count-two";
"00 מיליארד",
    "100000000000-count-many";
"00 מיליארד",
    "100000000000-count-other";
"00 מיליארד",
    "100000000000-count-one";
"0 טריליון",
    "1000000000000-count-two";
"0 טריליון",
    "1000000000000-count-many";
"0 טריליון",
    "1000000000000-count-other";
"0 טריליון",
    "1000000000000-count-one";
"00 טריליון",
    "10000000000000-count-two";
"00 טריליון",
    "10000000000000-count-many";
"00 טריליון",
    "10000000000000-count-other";
"00 טריליון",
    "10000000000000-count-one";
"000 טריליון",
    "100000000000000-count-two";
"000 טריליון",
    "100000000000000-count-many";
"000 טריליון",
    "100000000000000-count-other";
"000 טריליון",
    "100000000000000-count-one";
"000 טריליון",
    "100000000000000-count-many";
```

```
        "100000000000000-count-other";
        "100 טריליון 000";
    }
}
"short";
{
    "decimalFormat";
    {
        "1000-count-one";
        "0K",
        "1000-count-two";
        "0K",
        "1000-count-many";
        "0K",
        "1000-count-other";
        "0K",
        "10000-count-one";
        "00K",
        "10000-count-two";
        "00K",
        "10000-count-many";
        "00K",
        "10000-count-other";
        "00K",
        "100000-count-one";
        "000K",
        "100000-count-two";
        "000K",
        "100000-count-many";
        "000K",
        "100000-count-other";
        "000K",
        "1000000-count-one";
        "0M",
        "1000000-count-two";
        "0M",
        "1000000-count-many";
        "0M",
        "1000000-count-other";
        "0M",
        "10000000-count-one";
        "00M",
        "10000000-count-two";
        "00M",
        "10000000-count-many";
        "00M",
        "10000000-count-other";
        "00M",
        "100000000-count-one";
        "000M",
        "100000000-count-two";
        "000M",
        "100000000-count-many";
        "000M",
        "100000000-count-other";
        "000M",
        "1000000000-count-one";
    }
}
```

```

        "0B",
        "1000000000-count-two";
        "0B",
        "1000000000-count-many";
        "0B",
        "1000000000-count-other";
        "0B",
        "1000000000-count-one";
        "00B",
        "1000000000-count-two";
        "00B",
        "1000000000-count-many";
        "00B",
        "1000000000-count-other";
        "00B",
        "1000000000-count-one";
        "000B",
        "1000000000-count-two";
        "000B",
        "1000000000-count-many";
        "000B",
        "1000000000-count-other";
        "000B",
        "1000000000-count-one";
        "0T",
        "1000000000-count-two";
        "0T",
        "1000000000-count-many";
        "0T",
        "1000000000-count-other";
        "0T",
        "1000000000-count-one";
        "00T",
        "1000000000-count-two";
        "00T",
        "1000000000-count-many";
        "00T",
        "1000000000-count-other";
        "00T",
        "1000000000-count-one";
        "000T",
        "1000000000-count-two";
        "000T",
        "1000000000-count-many";
        "000T",
        "1000000000-count-other";
        "000T";
    }
}
}
"scientificFormats-numberSystem-latn";
{
    "standard";
    "#E0";
}
"percentFormats-numberSystem-latn";
{

```

```

        "standard";
        "#,##0%";
    }
    "currencyFormats-numberSystem-latn";
    {
        "currencySpacing";
        {
            "beforeCurrency";
            {
                "currencyMatch";
                "[:^S:]",
                "surroundingMatch";
                "[:digit:]",
                "insertBetween";
                " ";
            }
            "afterCurrency";
            {
                "currencyMatch";
                "[:^S:]",
                "surroundingMatch";
                "[:digit:]",
                "insertBetween";
                " ";
            }
        }
    }
    "standard";
    "##0.00#,-¤; ##0.00#, ¤",
    "accounting";
    "#,##0.00 ¤",
    "short";
    {
        "standard";
        {
            "1000-count-one";
            "¤ 0K",
            "1000-count-two";
            "¤ 0K",
            "1000-count-many";
            "¤0K",
            "1000-count-other";
            "¤ 0K",
            "10000-count-one";
            "¤00K",
            "10000-count-two";
            "¤00K",
            "10000-count-many";
            "¤00K",
            "10000-count-other";
            "¤ 00K",
            "100000-count-one";
            "¤000K",
            "100000-count-two";
            "¤000K",
            "100000-count-many";
            "¤000K",
            "100000-count-other";
        }
    }

```

```
"¤000K",
    "1000000-count-one";
"¤0M",
    "1000000-count-two";
"¤0M",
    "1000000-count-many";
"¤0M",
    "1000000-count-other";
"¤0M",
    "10000000-count-one";
"¤00M",
    "10000000-count-two";
"¤00M",
    "10000000-count-many";
"¤00M",
    "10000000-count-other";
"¤00M",
    "100000000-count-one";
"¤000M",
    "100000000-count-two";
"¤000M",
    "100000000-count-many";
"¤000M",
    "100000000-count-other";
"¤000M",
    "1000000000-count-one";
"¤0B",
    "1000000000-count-two";
"¤0B",
    "1000000000-count-many";
"¤0B",
    "1000000000-count-other";
"¤0B",
    "10000000000-count-one";
"¤00B",
    "10000000000-count-two";
"¤00B",
    "10000000000-count-many";
"¤00B",
    "10000000000-count-other";
"¤00B",
    "100000000000-count-one";
"¤000B",
    "100000000000-count-two";
"¤000B",
    "100000000000-count-many";
"¤000B",
    "100000000000-count-other";
"¤000B",
    "1000000000000-count-one";
"¤0T",
    "1000000000000-count-two";
"¤0T",
    "1000000000000-count-many";
"¤0T",
    "1000000000000-count-other";
"¤0T",
```

```

        "10000000000000-count-one";
        "א00T",
        "10000000000000-count-two";
        "א00T",
        "10000000000000-count-many";
        "א00T",
        "10000000000000-count-other";
        "א00T",
        "10000000000000-count-one";
        "א000T",
        "10000000000000-count-two";
        "א000T",
        "10000000000000-count-many";
        "א000T",
        "10000000000000-count-other";
        "א000T";
    }
}
"unitPattern-count-one";
"{0} {1}",
    "unitPattern-count-two";
"{0} {1}",
    "unitPattern-count-many";
"{0} {1}",
    "unitPattern-count-other";
"{0} {1}";
}
"miscPatterns-numberSystem-latn";
{
    "atLeast";
    "≥{0}+",
        "range";
    "{0}-{1}";
}
"minimalPairs";
{
    "pluralMinimalPairs";
    "שנה",
        "pluralMinimalPairs";
    "שנתיים",
        "pluralMinimalPairs";
    "{0} שנה",
        "pluralMinimalPairs";
    "{0} שנים",
        "other";
    "פנה-פנה ימינה בפנייה ה";
}
}
}
}

```

TIMEZONENAMES.JSON

```
{
  "main": {
```



```

"he": {
  "identity": {
    "version": {
      "_number": "$Revision: 13259 $",
      "_cldrVersion": "31.0.1"
    },
    "language": "he"
  },
  "dates": {
    "timeZoneNames": {
      "hourFormat": "+HH:mm;-HH:mm",
      "gmtFormat": "GMT{0}",
      "gmtZeroFormat": "GMT",
      "regionFormat": "שעון {0}",
      "regionFormat-type-daylight": "קיצ (0{ שעון",
      "regionFormat-type-standard": "חורף (0{ שעון",
      "fallbackFormat": "{1} ({0})",
      "zone": {
        "America": {
          "Adak": {
            "exemplarCity": "אדאק"
          },
          "Anchorage": {
            "exemplarCity": "אנקורג'י"
          },
          "Anguilla": {
            "exemplarCity": "אנגווילה"
          },
          "Antigua": {
            "exemplarCity": "אנטיגואה"
          },
          "Araguaina": {
            "exemplarCity": "אראגואינה"
          },
          "Argentina": {
            "Rio_Gallegos": {
              "exemplarCity": "ריו גאייגוס"
            },
            "San_Juan": {
              "exemplarCity": "סן חואן"
            },
            "Ushuaia": {
              "exemplarCity": "אושואיה"
            },
            "La_Rioja": {
              "exemplarCity": "לה ריוחה"
            },
            "San_Luis": {
              "exemplarCity": "סן לואיס"
            },
            "Salta": {
              "exemplarCity": "סלטה"
            },
            "Tucuman": {
              "exemplarCity": "טוקומן"
            }
          }
        }
      }
    }
  }
}

```

```
"Aruba": {
  "exemplarCity": "ארובה"
},
"Asuncion": {
  "exemplarCity": "אסונסיון"
},
"Bahia": {
  "exemplarCity": "באהיה"
},
"Bahia_Banderas": {
  "exemplarCity": "באהיה בנדרס"
},
"Barbados": {
  "exemplarCity": "ברבדוס"
},
"Belem": {
  "exemplarCity": "בלם"
},
"Belize": {
  "exemplarCity": "בליז"
},
"Blanc-Sablon": {
  "exemplarCity": "בלאן-סבלון"
},
"Boa_Vista": {
  "exemplarCity": "בואה ויסטה"
},
"Bogota": {
  "exemplarCity": "בוגוטה"
},
"Boise": {
  "exemplarCity": "בויסי"
},
"Buenos_Aires": {
  "exemplarCity": "בואנוס איירס"
},
"Cambridge_Bay": {
  "exemplarCity": "קיימברידג' ביי"
},
"Campo_Grande": {
  "exemplarCity": "קמפו גרנדה"
},
"Cancun": {
  "exemplarCity": "קנקון"
},
"Caracas": {
  "exemplarCity": "קראקס"
},
"Catamarca": {
  "exemplarCity": "קטמרקה"
},
"Cayenne": {
  "exemplarCity": "קאייין"
},
"Cayman": {
  "exemplarCity": "קיימן"
},
```

```
"Chicago": {
  "exemplarCity": "שיקגו"
},
"Chihuahua": {
  "exemplarCity": "צ'יוואוואה"
},
"Coral_Harbour": {
  "exemplarCity": "אטיקוקן"
},
"Cordoba": {
  "exemplarCity": "קורדובה"
},
"Costa_Rica": {
  "exemplarCity": "קוסטה ריקה"
},
"Creston": {
  "exemplarCity": "קרסטון"
},
"Cuiaba": {
  "exemplarCity": "קויאבה"
},
"Curacao": {
  "exemplarCity": "קוראסאו"
},
"Danmarkshavn": {
  "exemplarCity": "דנמרקסהוון"
},
"Dawson": {
  "exemplarCity": "דוסון"
},
"Dawson_Creek": {
  "exemplarCity": "דוסון קריק"
},
"Denver": {
  "exemplarCity": "דנוור"
},
"Detroit": {
  "exemplarCity": "דטרויט"
},
"Dominica": {
  "exemplarCity": "דומיניקה"
},
"Edmonton": {
  "exemplarCity": "אדמונטון"
},
"Eirunepe": {
  "exemplarCity": "אירונפי"
},
"El_Salvador": {
  "exemplarCity": "אל סלבדור"
},
"Fort_Nelson": {
  "exemplarCity": "פורט נלסון"
},
"Fortaleza": {
  "exemplarCity": "פורטאלזה"
},
},
```

```
"Glace_Bay": {
  "exemplarCity": "גלייס ביי"
},
"Godthab": {
  "exemplarCity": "נואוק"
},
"Goose_Bay": {
  "exemplarCity": "גוס ביי"
},
"Grand_Turk": {
  "exemplarCity": "גרנד טורק"
},
"Grenada": {
  "exemplarCity": "גרנדה"
},
"Guadeloupe": {
  "exemplarCity": "גואדלופ"
},
"Guatemala": {
  "exemplarCity": "גואטמלה"
},
"Guayaquil": {
  "exemplarCity": "גואיאקיל"
},
"Guyana": {
  "exemplarCity": "גיאנה"
},
"Halifax": {
  "exemplarCity": "הליפקס"
},
"Havana": {
  "exemplarCity": "הוואנה"
},
"Hermosillo": {
  "exemplarCity": "הרמוסיו"
},
"Indiana": {
  "Vincennes": {
    "exemplarCity": "וינסנס, אינדיאנה"
  },
  "Petersburg": {
    "exemplarCity": "פיטרסבורג, אינדיאנה"
  },
  "Tell_City": {
    "exemplarCity": "טל סיטי, אינדיאנה"
  },
  "Knox": {
    "exemplarCity": "נוקס, אינדיאנה"
  },
  "Winamac": {
    "exemplarCity": "וינמאק, אינדיאנה"
  },
  "Marengo": {
    "exemplarCity": "מרנגו, אינדיאנה"
  },
  "Vevay": {
    "exemplarCity": "ויוואיי, אינדיאנה"
  }
}
```

```

    },
    "Indianapolis": {
      "exemplarCity": "אינדיאנפוליס"
    },
    "Inuvik": {
      "exemplarCity": "אינוויק"
    },
    "Iqaluit": {
      "exemplarCity": "איקלואיט"
    },
    "Jamaica": {
      "exemplarCity": "ג'מייקה"
    },
    "Jujuy": {
      "exemplarCity": "חוחוי"
    },
    "Juneau": {
      "exemplarCity": "ג'וננו"
    },
    "Kentucky": {
      "Monticello": {
        "exemplarCity": "מונטיצ'לו, קנטאקי"
      }
    },
    "Kralendijk": {
      "exemplarCity": "קרלנדייק"
    },
    "La_Paz": {
      "exemplarCity": "לה פאס"
    },
    "Lima": {
      "exemplarCity": "לימה"
    },
    "Los_Angeles": {
      "exemplarCity": "לוס אנג'לס"
    },
    "Louisville": {
      "exemplarCity": "לואיוויל"
    },
    "Lower_Princes": {
      "exemplarCity": "לואוור פרינסס קוורטר"
    },
    "Maceio": {
      "exemplarCity": "מסיאו"
    },
    "Managua": {
      "exemplarCity": "מנגואה"
    },
    "Manaus": {
      "exemplarCity": "מנאוס"
    },
    "Marigot": {
      "exemplarCity": "מריגו"
    },
    "Martinique": {
      "exemplarCity": "מרטיניק"
    }
  }

```

```
    },  
    "Matamoros": {  
      "exemplarCity": "מטמורוס"  
    },  
    "Mazatlan": {  
      "exemplarCity": "מזטלן"  
    },  
    "Mendoza": {  
      "exemplarCity": "מנדוזה"  
    },  
    "Menominee": {  
      "exemplarCity": "מנומיני"  
    },  
    "Merida": {  
      "exemplarCity": "מרידה"  
    },  
    "Metlakatla": {  
      "exemplarCity": "מטלקטלה"  
    },  
    "Mexico_City": {  
      "exemplarCity": "מקסיקו סיטי"  
    },  
    "Miquelon": {  
      "exemplarCity": "מיקלון"  
    },  
    "Moncton": {  
      "exemplarCity": "מונקטון"  
    },  
    "Monterrey": {  
      "exemplarCity": "מונטריי"  
    },  
    "Montevideo": {  
      "exemplarCity": "מונטווידאו"  
    },  
    "Montserrat": {  
      "exemplarCity": "מונטסראט"  
    },  
    "Nassau": {  
      "exemplarCity": "נסאו"  
    },  
    "New_York": {  
      "exemplarCity": "ניו יורק"  
    },  
    "Nipigon": {  
      "exemplarCity": "ניפיגון"  
    },  
    "Nome": {  
      "exemplarCity": "נום"  
    },  
    "Noronha": {  
      "exemplarCity": "נורוניה"  
    },  
    "North_Dakota": {  
      "Beulah": {  
        "exemplarCity": "ביולה, צפון דקוטה"  
      },  
      "New_Salem": {
```

```

        "exemplarCity": "ניו סיילס, צפון דקוטה"
    },
    "Center": {
        "exemplarCity": "סנטר, צפון דקוטה"
    }
},
"Ojinaga": {
    "exemplarCity": "אוג'ינאגה"
},
"Panama": {
    "exemplarCity": "פנמה"
},
"Pangnirtung": {
    "exemplarCity": "פנגנירטונג"
},
"Paramaribo": {
    "exemplarCity": "פרמריבו"
},
"Phoenix": {
    "exemplarCity": "פיניקס"
},
"Port-au-Prince": {
    "exemplarCity": "פורט או פראנס"
},
"Port_of_Spain": {
    "exemplarCity": "פורט אוף ספייין"
},
"Porto_Velho": {
    "exemplarCity": "פורטו וליו"
},
"Puerto_Rico": {
    "exemplarCity": "פוארטו ריקו"
},
"Rainy_River": {
    "exemplarCity": "רייני ריבר"
},
"Rankin_Inlet": {
    "exemplarCity": "רנקין אינלט"
},
"Recife": {
    "exemplarCity": "רסיפה"
},
"Regina": {
    "exemplarCity": "רג'ינה"
},
"Resolute": {
    "exemplarCity": "רזולוט"
},
"Rio_Branco": {
    "exemplarCity": "ריו ברנקו"
},
"Santa_Isabel": {
    "exemplarCity": "סנטה איסבל"
},
"Santarem": {
    "exemplarCity": "סנטרם"
},
},

```

```
"Santiago": {
  "exemplarCity": "סנטיאגו"
},
"Santo_Domingo": {
  "exemplarCity": "סנטו דומינגו"
},
"Sao_Paulo": {
  "exemplarCity": "סאו פאולו"
},
"Scoresbysund": {
  "exemplarCity": "סקורסביסונד"
},
"Sitka": {
  "exemplarCity": "סיטקה"
},
"St_Barthelemy": {
  "exemplarCity": "סנט ברתלמי"
},
"St_Johns": {
  "exemplarCity": "סנט ג'ונס"
},
"St_Kitts": {
  "exemplarCity": "סנט קיטס"
},
"St_Lucia": {
  "exemplarCity": "סנט לוסיה"
},
"St_Thomas": {
  "exemplarCity": "סנט תומאס"
},
"St_Vincent": {
  "exemplarCity": "סנט וינסנט"
},
"Swift_Current": {
  "exemplarCity": "סוויפט קרנט"
},
"Tegucigalpa": {
  "exemplarCity": "טגוסיגלפה"
},
"Thule": {
  "exemplarCity": "תולה"
},
"Thunder_Bay": {
  "exemplarCity": "ת'אנדר ביי"
},
"Tijuana": {
  "exemplarCity": "טיחואנה"
},
"Toronto": {
  "exemplarCity": "טורונטו"
},
"Tortola": {
  "exemplarCity": "טורטולה"
},
"Vancouver": {
  "exemplarCity": "ונקובר"
},
},
```



```
"Whitehorse": {
  "exemplarCity": "ווייטהורס"
},
"Winnipeg": {
  "exemplarCity": "וויניפג"
},
"Yakutat": {
  "exemplarCity": "יקוטאט"
},
"Yellowknife": {
  "exemplarCity": "ילונייף"
}
},
"Atlantic": {
  "Azores": {
    "exemplarCity": "האיים האזוריים"
  },
  "Bermuda": {
    "exemplarCity": "ברמודה"
  },
  "Canary": {
    "exemplarCity": "האיים הקנריים"
  },
  "Cape_Verde": {
    "exemplarCity": "כף ורדה"
  },
  "Faeroe": {
    "exemplarCity": "פארו"
  },
  "Madeira": {
    "exemplarCity": "מדיירה"
  },
  "Reykjavik": {
    "exemplarCity": "רייקיאוויק"
  },
  "South_Georgia": {
    "exemplarCity": "דרום ג'ורג'יה"
  },
  "St_Helena": {
    "exemplarCity": "סנט הלנה"
  },
  "Stanley": {
    "exemplarCity": "סטנלי"
  }
},
"Europe": {
  "Amsterdam": {
    "exemplarCity": "אמסטרדם"
  },
  "Andorra": {
    "exemplarCity": "אנדורה"
  },
  "Astrakhan": {
    "exemplarCity": "אסטרקחן"
  },
  "Athens": {
    "exemplarCity": "אתונה"
```

```
},
"Belgrade": {
  "exemplarCity": "בלגרד"
},
"Berlin": {
  "exemplarCity": "ברלין"
},
"Bratislava": {
  "exemplarCity": "ברטיסלבה"
},
"Brussels": {
  "exemplarCity": "בריסל"
},
"Bucharest": {
  "exemplarCity": "בוקרשט"
},
"Budapest": {
  "exemplarCity": "בודפשט"
},
"Busingen": {
  "exemplarCity": "ביזינגן"
},
"Chisinau": {
  "exemplarCity": "קיישינב"
},
"Copenhagen": {
  "exemplarCity": "קופנהגן"
},
"Dublin": {
  "long": {
    "daylight": "שעון קיץ אירלנד"
  },
  "exemplarCity": "דבלין"
},
"Gibraltar": {
  "exemplarCity": "גיברלטר"
},
"Guernsey": {
  "exemplarCity": "גרנזי"
},
"Helsinki": {
  "exemplarCity": "הלסינקי"
},
"Isle_of_Man": {
  "exemplarCity": "האי מאן"
},
"Istanbul": {
  "exemplarCity": "איסטנבול"
},
"Jersey": {
  "exemplarCity": "ג'רזי"
},
"Kaliningrad": {
  "exemplarCity": "קלינינגרד"
},
"Kiev": {
  "exemplarCity": "קייב"
}
```

```
    },
    "Kirov": {
      "exemplarCity": "קירוב"
    },
    "Lisbon": {
      "exemplarCity": "ליסבון"
    },
    "Ljubljana": {
      "exemplarCity": "לובליאנה"
    },
    "London": {
      "long": {
        "daylight": "שעון קיץ בריטניה"
      },
      "exemplarCity": "לונדון"
    },
    "Luxembourg": {
      "exemplarCity": "לוקסמבורג"
    },
    "Madrid": {
      "exemplarCity": "מדריד"
    },
    "Malta": {
      "exemplarCity": "מלטה"
    },
    "Mariehamn": {
      "exemplarCity": "מריהאמן"
    },
    "Minsk": {
      "exemplarCity": "מינסק"
    },
    "Monaco": {
      "exemplarCity": "מונקו"
    },
    "Moscow": {
      "exemplarCity": "מוסקבה"
    },
    "Oslo": {
      "exemplarCity": "אוסלו"
    },
    "Paris": {
      "exemplarCity": "פריז"
    },
    "Podgorica": {
      "exemplarCity": "פודגוריצה"
    },
    "Prague": {
      "exemplarCity": "פראג"
    },
    "Riga": {
      "exemplarCity": "ריגה"
    },
    "Rome": {
      "exemplarCity": "רומא"
    },
    "Samara": {
      "exemplarCity": "סמרה"
    }
  }
```

```
    },  
    "San_Marino": {  
      "exemplarCity": "סן מרינו"  
    },  
    "Sarajevo": {  
      "exemplarCity": "סרייבו"  
    },  
    "Simferopol": {  
      "exemplarCity": "סימפרופול"  
    },  
    "Skopje": {  
      "exemplarCity": "סקופיה"  
    },  
    "Sofia": {  
      "exemplarCity": "סופיה"  
    },  
    "Stockholm": {  
      "exemplarCity": "שטוקהולם"  
    },  
    "Tallinn": {  
      "exemplarCity": "טאלין"  
    },  
    "Tirane": {  
      "exemplarCity": "טירנה"  
    },  
    "Ulyanovsk": {  
      "exemplarCity": "אוליאנובסק"  
    },  
    "Uzhgorod": {  
      "exemplarCity": "אוז'הורוד"  
    },  
    "Vaduz": {  
      "exemplarCity": "ואדוץ"  
    },  
    "Vatican": {  
      "exemplarCity": "הוותיקן"  
    },  
    "Vienna": {  
      "exemplarCity": "וינה"  
    },  
    "Vilnius": {  
      "exemplarCity": "וילנה"  
    },  
    "Volgograd": {  
      "exemplarCity": "וולגוגרד"  
    },  
    "Warsaw": {  
      "exemplarCity": "ורשה"  
    },  
    "Zagreb": {  
      "exemplarCity": "זאגרב"  
    },  
    "Zaporozhye": {  
      "exemplarCity": "זפורוז'יה"  
    },  
    "Zurich": {  
      "exemplarCity": "ציריך"
```

```
    }  
  },  
  "Africa": {  
    "Abidjan": {  
      "exemplarCity": "אביג'אן"  
    },  
    "Accra": {  
      "exemplarCity": "אקרה"  
    },  
    "Addis_Ababa": {  
      "exemplarCity": "אדיס אבבה"  
    },  
    "Algiers": {  
      "exemplarCity": "אלג'יר"  
    },  
    "Asmera": {  
      "exemplarCity": "אסמרה"  
    },  
    "Bamako": {  
      "exemplarCity": "במאקו"  
    },  
    "Bangui": {  
      "exemplarCity": "בנגווי"  
    },  
    "Banjul": {  
      "exemplarCity": "בנג'ול"  
    },  
    "Bissau": {  
      "exemplarCity": "ביסאו"  
    },  
    "Blantyre": {  
      "exemplarCity": "בלנטיר"  
    },  
    "Brazzaville": {  
      "exemplarCity": "ברזוויל"  
    },  
    "Bujumbura": {  
      "exemplarCity": "בוג'ומבורה"  
    },  
    "Cairo": {  
      "exemplarCity": "קהיר"  
    },  
    "Casablanca": {  
      "exemplarCity": "קזבלנקה"  
    },  
    "Ceuta": {  
      "exemplarCity": "סאוטה"  
    },  
    "Conakry": {  
      "exemplarCity": "קונאקרי"  
    },  
    "Dakar": {  
      "exemplarCity": "דקאר"  
    },  
    "Dar_es_Salaam": {  
      "exemplarCity": "דאר א-סלאם"  
    },  
  },  
}
```

```
"Djibouti": {
  "exemplarCity": "ג'יבוטי"
},
"Douala": {
  "exemplarCity": "דואלה"
},
"El_Aaiun": {
  "exemplarCity": "אל עיון"
},
"Freetown": {
  "exemplarCity": "פריטאון"
},
"Gaborone": {
  "exemplarCity": "גבורונה"
},
"Harare": {
  "exemplarCity": "הרארה"
},
"Johannesburg": {
  "exemplarCity": "יוהנסבורג"
},
"Juba": {
  "exemplarCity": "ג'ובה"
},
"Kampala": {
  "exemplarCity": "קמפלה"
},
"Khartoum": {
  "exemplarCity": "חרטום"
},
"Kigali": {
  "exemplarCity": "קיגלי"
},
"Kinshasa": {
  "exemplarCity": "קינשסה"
},
"Lagos": {
  "exemplarCity": "לגוס"
},
"Libreville": {
  "exemplarCity": "ליברוויל"
},
"Lome": {
  "exemplarCity": "לומה"
},
"Luanda": {
  "exemplarCity": "לואנדה"
},
"Lubumbashi": {
  "exemplarCity": "לובומבאשי"
},
"Lusaka": {
  "exemplarCity": "לוסקה"
},
"Malabo": {
  "exemplarCity": "מלבו"
},
}
```

```

    "Maputo": {
      "exemplarCity": "מאפוטו"
    },
    "Maseru": {
      "exemplarCity": "מסרו"
    },
    "Mbabane": {
      "exemplarCity": "אמבאבאנה"
    },
    "Mogadishu": {
      "exemplarCity": "מוגדישו"
    },
    "Monrovia": {
      "exemplarCity": "מונרוביה"
    },
    "Nairobi": {
      "exemplarCity": "ניירובי"
    },
    "Ndjamena": {
      "exemplarCity": "נג'מנה"
    },
    "Niamey": {
      "exemplarCity": "ניאמי"
    },
    "Nouakchott": {
      "exemplarCity": "נואקצ'וט"
    },
    "Ouagadougou": {
      "exemplarCity": "וואגאדוגו"
    },
    "Porto-Novo": {
      "exemplarCity": "פורטו נובו"
    },
    "Sao_Tome": {
      "exemplarCity": "סאו טומה"
    },
    "Tripoli": {
      "exemplarCity": "טריפולי"
    },
    "Tunis": {
      "exemplarCity": "תוניס"
    },
    "Windhoek": {
      "exemplarCity": "וינדהוק"
    }
  },
  "Asia": {
    "Aden": {
      "exemplarCity": "עדן"
    },
    "Almaty": {
      "exemplarCity": "אלמאטי"
    },
    "Amman": {
      "exemplarCity": "עמאן"
    },
    "Anadyr": {

```

```
    "exemplarCity": "אנדיר"
  },
  "Aqtau": {
    "exemplarCity": "אקטאוו"
  },
  "Aqtobe": {
    "exemplarCity": "אקטובה"
  },
  "Ashgabat": {
    "exemplarCity": "אשגבט"
  },
  "Baghdad": {
    "exemplarCity": "בגדד"
  },
  "Bahrain": {
    "exemplarCity": "בחריין"
  },
  "Baku": {
    "exemplarCity": "באקו"
  },
  "Bangkok": {
    "exemplarCity": "בנגקוק"
  },
  "Barnaul": {
    "exemplarCity": "ברנאול"
  },
  "Beirut": {
    "exemplarCity": "ביירות"
  },
  "Bishkek": {
    "exemplarCity": "בישקק"
  },
  "Brunei": {
    "exemplarCity": "ברוניי"
  },
  "Calcutta": {
    "exemplarCity": "קולקטה"
  },
  "Chita": {
    "exemplarCity": "צ'יטה"
  },
  "Choibalsan": {
    "exemplarCity": "צ'ויבלסן"
  },
  "Colombo": {
    "exemplarCity": "קולומבו"
  },
  "Damascus": {
    "exemplarCity": "דמשק"
  },
  "Dhaka": {
    "exemplarCity": "דאקה"
  },
  "Dili": {
    "exemplarCity": "דילי"
  },
  "Dubai": {
```



```
    "exemplarCity": "דובאי"
  },
  "Dushanbe": {
    "exemplarCity": "דושנבה"
  },
  "Gaza": {
    "exemplarCity": "עזה"
  },
  "Hebron": {
    "exemplarCity": "חברון"
  },
  "Hong_Kong": {
    "exemplarCity": "הונג קונג"
  },
  "Hovd": {
    "exemplarCity": "חובד"
  },
  "Irkutsk": {
    "exemplarCity": "אירקוטסק"
  },
  "Jakarta": {
    "exemplarCity": "ג'קרטה"
  },
  "Jayapura": {
    "exemplarCity": "ג'איאפורה"
  },
  "Jerusalem": {
    "exemplarCity": "ירושלים"
  },
  "Kabul": {
    "exemplarCity": "קאבול"
  },
  "Kamchatka": {
    "exemplarCity": "קמצ'טקה"
  },
  "Karachi": {
    "exemplarCity": "קראצ'י"
  },
  "Katmandu": {
    "exemplarCity": "קטמנדו"
  },
  "Khandyga": {
    "exemplarCity": "חנדיגה"
  },
  "Krasnoyarsk": {
    "exemplarCity": "קרסנויארסק"
  },
  "Kuala_Lumpur": {
    "exemplarCity": "קואלה לומפור"
  },
  "Kuching": {
    "exemplarCity": "קוצ'ינג"
  },
  "Kuwait": {
    "exemplarCity": "כווית"
  },
  "Macau": {
```

```

    "exemplarCity": "מקאו"
  },
  "Magadan": {
    "exemplarCity": "מגדן"
  },
  "Makassar": {
    "exemplarCity": "מאקאסאר"
  },
  "Manila": {
    "exemplarCity": "מנילה"
  },
  "Muscat": {
    "exemplarCity": "מוסקט"
  },
  "Nicosia": {
    "exemplarCity": "ניקוסיה"
  },
  "Novokuznetsk": {
    "exemplarCity": "נובוקוזנטסק"
  },
  "Novosibirsk": {
    "exemplarCity": "נובוסיבירסק"
  },
  "Omsk": {
    "exemplarCity": "אומסק"
  },
  "Oral": {
    "exemplarCity": "אורל"
  },
  "Phnom_Penh": {
    "exemplarCity": "פנום פן"
  },
  "Pontianak": {
    "exemplarCity": "פונטיאנק"
  },
  "Pyongyang": {
    "exemplarCity": "פיונגיאנג"
  },
  "Qatar": {
    "exemplarCity": "קטאר"
  },
  "Qyzylorda": {
    "exemplarCity": "קיזילורדה"
  },
  "Rangoon": {
    "exemplarCity": "רנגון"
  },
  "Riyadh": {
    "exemplarCity": "ריאד"
  },
  "Saigon": {
    "exemplarCity": "הו צ'י מין סיטי"
  },
  "Sakhalin": {
    "exemplarCity": "סחלין"
  },
  "Samarkand": {

```

```
    "exemplarCity": "סמרקנד"
  },
  "Seoul": {
    "exemplarCity": "סיאול"
  },
  "Shanghai": {
    "exemplarCity": "שנחאי"
  },
  "Singapore": {
    "exemplarCity": "סינגפור"
  },
  "Srednekolymsk": {
    "exemplarCity": "סרדנייקולימסק"
  },
  "Taipei": {
    "exemplarCity": "טאיפיי"
  },
  "Tashkent": {
    "exemplarCity": "טשקנט"
  },
  "Tbilisi": {
    "exemplarCity": "טביליסי"
  },
  "Tehran": {
    "exemplarCity": "טהרן"
  },
  "Thimphu": {
    "exemplarCity": "טהימפהו"
  },
  "Tokyo": {
    "exemplarCity": "טוקיו"
  },
  "Tomsk": {
    "exemplarCity": "טומסק"
  },
  "Ulaanbaatar": {
    "exemplarCity": "אולאאנבטאר"
  },
  "Urumqi": {
    "exemplarCity": "אורומקי"
  },
  "Ust-Nera": {
    "exemplarCity": "אוסט-נרה"
  },
  "Vientiane": {
    "exemplarCity": "האנוי"
  },
  "Vladivostok": {
    "exemplarCity": "ולדיווסטוק"
  },
  "Yakutsk": {
    "exemplarCity": "יקוטסק"
  },
  "Yekaterinburg": {
    "exemplarCity": "יקטרינבורג"
  },
  "Yerevan": {
```

```

        "exemplarCity": "ירושלם"
    },
    },
    "Indian": {
        "Antananarivo": {
            "exemplarCity": "אנטננריבו"
        },
        "Chagos": {
            "exemplarCity": "צ'אגוס"
        },
        "Christmas": {
            "exemplarCity": "האי כריסטמס"
        },
        "Cocos": {
            "exemplarCity": "קוקוס"
        },
        "Comoro": {
            "exemplarCity": "קומורו"
        },
        "Kerguelen": {
            "exemplarCity": "קרגוולן"
        },
        "Mahe": {
            "exemplarCity": "מהא"
        },
        "Maldives": {
            "exemplarCity": "האיים המלדיביים"
        },
        "Mauritius": {
            "exemplarCity": "מאוריציוס"
        },
        "Mayotte": {
            "exemplarCity": "מאיוט"
        },
        "Reunion": {
            "exemplarCity": "ראוניון"
        }
    },
    "Australia": {
        "Adelaide": {
            "exemplarCity": "אדלייד"
        },
        "Brisbane": {
            "exemplarCity": "בריסביין"
        },
        "Broken_Hill": {
            "exemplarCity": "ברוקן היל"
        },
        "Currie": {
            "exemplarCity": "קרי"
        },
        "Darwin": {
            "exemplarCity": "דרוויין"
        },
        "Eucla": {
            "exemplarCity": "יוקלה"
        }
    },
    },

```

```
"Hobart": {
  "exemplarCity": "הוברט"
},
"Lindeman": {
  "exemplarCity": "לינדמן"
},
"Lord_Howe": {
  "exemplarCity": "אי הלורד האו"
},
"Melbourne": {
  "exemplarCity": "מלבורן"
},
"Perth": {
  "exemplarCity": "פרת'"
},
"Sydney": {
  "exemplarCity": "סידני"
}
},
"Pacific": {
  "Apia": {
    "exemplarCity": "אפיה"
  },
  "Auckland": {
    "exemplarCity": "אוקלנד"
  },
  "Bougainville": {
    "exemplarCity": "בוגנוויל"
  },
  "Chatham": {
    "exemplarCity": "צ'אטהאם"
  },
  "Easter": {
    "exemplarCity": "אי הפסחא"
  },
  "Efate": {
    "exemplarCity": "אפטה"
  },
  "Enderbury": {
    "exemplarCity": "אנדרבורי"
  },
  "Fakaofu": {
    "exemplarCity": "פקאופו"
  },
  "Fiji": {
    "exemplarCity": "פיג'י"
  },
  "Funafuti": {
    "exemplarCity": "פונפוט"
  },
  "Galapagos": {
    "exemplarCity": "גלפאגוס"
  },
  "Gambier": {
    "exemplarCity": "איי גמבייה"
  },
  "Guadalcanal": {
```

```
    "exemplarCity": "גוודלוקנאל"
  },
  "Guam": {
    "exemplarCity": "גואם"
  },
  "Honolulu": {
    "exemplarCity": "הונוולולו"
  },
  "Johnston": {
    "exemplarCity": "ג'ונסטון"
  },
  "Kiritimati": {
    "exemplarCity": "קיריטימאטי"
  },
  "Kosrae": {
    "exemplarCity": "קוסרה"
  },
  "Kwajalein": {
    "exemplarCity": "קוואג'ליין"
  },
  "Majuro": {
    "exemplarCity": "מאג'ורו"
  },
  "Marquesas": {
    "exemplarCity": "איי מרקייז"
  },
  "Midway": {
    "exemplarCity": "מידוויי"
  },
  "Nauru": {
    "exemplarCity": "נאורו"
  },
  "Niue": {
    "exemplarCity": "ניואה"
  },
  "Norfolk": {
    "exemplarCity": "נורפוק"
  },
  "Noumea": {
    "exemplarCity": "נומאה"
  },
  "Pago_Pago": {
    "exemplarCity": "פאגו פאגו"
  },
  "Palau": {
    "exemplarCity": "פלאו"
  },
  "Pitcairn": {
    "exemplarCity": "פיטקרן"
  },
  "Ponape": {
    "exemplarCity": "פונפיי"
  },
  "Port_Moresby": {
    "exemplarCity": "פורט מורסבי"
  },
  "Rarotonga": {
```

```

        "exemplarCity": "רארוטונגה"
      },
      "Saipan": {
        "exemplarCity": "סאיפאן"
      },
      "Tahiti": {
        "exemplarCity": "טהיטי"
      },
      "Tarawa": {
        "exemplarCity": "טאראווה"
      },
      "Tongatapu": {
        "exemplarCity": "טונגטאפו"
      },
      "Truk": {
        "exemplarCity": "צ'וק"
      },
      "Wake": {
        "exemplarCity": "ווייק"
      },
      "Wallis": {
        "exemplarCity": "ווליס"
      }
    },
    "Arctic": {
      "Longyearbyen": {
        "exemplarCity": "לונגיירבין"
      }
    },
    "Antarctica": {
      "Casey": {
        "exemplarCity": "קאסיי"
      },
      "Davis": {
        "exemplarCity": "דייוויס"
      },
      "DumontDUrville": {
        "exemplarCity": "דומון ד'אורוויל"
      },
      "Macquarie": {
        "exemplarCity": "מקרי"
      },
      "Mawson": {
        "exemplarCity": "מוסון"
      },
      "McMurdo": {
        "exemplarCity": "מק-מרדו"
      },
      "Palmer": {
        "exemplarCity": "פאלמר"
      },
      "Rothera": {
        "exemplarCity": "רות'רה"
      },
      "Syowa": {
        "exemplarCity": "סיוואה"
      }
    },
  },

```

```
"Troll": {
  "exemplarCity": "טרול"
},
"Vostok": {
  "exemplarCity": "ווסטוק"
}
},
"Etc": {
  "GMT": {
    "exemplarCity": "GMT"
  },
  "GMT1": {
    "exemplarCity": "GMT+1"
  },
  "GMT10": {
    "exemplarCity": "GMT+10"
  },
  "GMT11": {
    "exemplarCity": "GMT+11"
  },
  "GMT12": {
    "exemplarCity": "GMT+12"
  },
  "GMT2": {
    "exemplarCity": "GMT+2"
  },
  "GMT3": {
    "exemplarCity": "GMT+3"
  },
  "GMT4": {
    "exemplarCity": "GMT+4"
  },
  "GMT5": {
    "exemplarCity": "GMT+5"
  },
  "GMT6": {
    "exemplarCity": "GMT+6"
  },
  "GMT7": {
    "exemplarCity": "GMT+7"
  },
  "GMT8": {
    "exemplarCity": "GMT+8"
  },
  "GMT9": {
    "exemplarCity": "GMT+9"
  },
  "GMT-1": {
    "exemplarCity": "GMT-1"
  },
  "GMT-10": {
    "exemplarCity": "GMT-10"
  },
  "GMT-11": {
    "exemplarCity": "GMT-11"
  },
  "GMT-12": {
```



```

        "exemplarCity": "GMT-12"
      },
      "GMT-13": {
        "exemplarCity": "GMT-13"
      },
      "GMT-14": {
        "exemplarCity": "GMT-14"
      },
      "GMT-2": {
        "exemplarCity": "GMT-2"
      },
      "GMT-3": {
        "exemplarCity": "GMT-3"
      },
      "GMT-4": {
        "exemplarCity": "GMT-4"
      },
      "GMT-5": {
        "exemplarCity": "GMT-5"
      },
      "GMT-6": {
        "exemplarCity": "GMT-6"
      },
      "GMT-7": {
        "exemplarCity": "GMT-7"
      },
      "GMT-8": {
        "exemplarCity": "GMT-8"
      },
      "GMT-9": {
        "exemplarCity": "GMT-9"
      },
      "UTC": {
        "long": {
          "standard": "זמן אוניברסלי מתואם"
        },
        "short": {
          "standard": "UTC"
        },
        "exemplarCity": "UTC"
      },
      "Unknown": {
        "exemplarCity": "עיר לא ידועה"
      }
    },
    "metazone": {
      "Afghanistan": {
        "long": {
          "standard": "שעון אפגניסטן"
        }
      },
      "Africa_Central": {
        "long": {
          "standard": "שעון מרכז אפריקה"
        }
      }
    },
  },

```

```

"Africa_Eastern": {
  "long": {
    "standard": "שעון מזרח אפריקה"
  }
},
"Africa_Southern": {
  "long": {
    "standard": "שעון דרום אפריקה"
  }
},
"Africa_Western": {
  "long": {
    "generic": "שעון מערב אפריקה",
    "standard": "שעון מערב אפריקה (חורף)",
    "daylight": "שעון מערב אפריקה (קיץ)"
  }
},
"Alaska": {
  "long": {
    "generic": "שעון אלסקה",
    "standard": "שעון אלסקה (חורף)",
    "daylight": "שעון אלסקה (קיץ)"
  }
},
"Amazon": {
  "long": {
    "generic": "שעון אמזונס",
    "standard": "שעון אמזונס (חורף)",
    "daylight": "שעון אמזונס (קיץ)"
  }
},
"America_Central": {
  "long": {
    "generic": "שעון מרכז ארה"ב",
    "standard": "שעון מרכז ארה"ב (חורף)",
    "daylight": "שעון מרכז ארה"ב (קיץ)"
  }
},
"America_Eastern": {
  "long": {
    "generic": "שעון החוף המזרחי",
    "standard": "שעון החוף המזרחי (חורף)",
    "daylight": "שעון החוף המזרחי (קיץ)"
  }
},
"America_Mountain": {
  "long": {
    "generic": "שעון אזור ההרים בארה"ב",
    "standard": "שעון אזור ההרים בארה"ב (חורף)",
    "daylight": "שעון אזור ההרים בארה"ב (קיץ)"
  }
},
"America_Pacific": {
  "long": {
    "generic": "שעון מערב ארה"ב",
    "standard": "שעון מערב ארה"ב (חורף)",
    "daylight": "שעון מערב ארה"ב (קיץ)"
  }
}

```

```

    },
    "Anadyr": {
      "long": {
        "generic": "שעון אנדיר",
        "standard": "שעון רגיל אנדיר",
        "daylight": "שעון קיץ אנדיר"
      }
    },
    "Apia": {
      "long": {
        "generic": "שעון אפיה",
        "standard": "שעון אפיה (חורף)",
        "daylight": "שעון אפיה (קיץ)"
      }
    },
    "Arabian": {
      "long": {
        "generic": "שעון חצי האי ערב",
        "standard": "שעון חצי האי ערב (חורף)",
        "daylight": "שעון חצי האי ערב (קיץ)"
      }
    },
    "Argentina": {
      "long": {
        "generic": "שעון ארגנטינה",
        "standard": "שעון ארגנטינה (חורף)",
        "daylight": "שעון ארגנטינה (קיץ)"
      }
    },
    "Argentina_Western": {
      "long": {
        "generic": "שעון מערב ארגנטינה",
        "standard": "שעון מערב ארגנטינה (חורף)",
        "daylight": "שעון מערב ארגנטינה (קיץ)"
      }
    },
    "Armenia": {
      "long": {
        "generic": "שעון ארמניה",
        "standard": "שעון ארמניה (חורף)",
        "daylight": "שעון ארמניה (קיץ)"
      }
    },
    "Atlantic": {
      "long": {
        "generic": "שעון האוקיינוס האטלנטי",
        "standard": "שעון האוקיינוס האטלנטי (חורף)",
        "daylight": "שעון האוקיינוס האטלנטי (קיץ)"
      }
    },
    "Australia_Central": {
      "long": {
        "generic": "שעון מרכז אוסטרליה",
        "standard": "שעון מרכז אוסטרליה (חורף)",
        "daylight": "שעון מרכז אוסטרליה (קיץ)"
      }
    }
  }

```

```

    },
    "Australia_CentralWestern": {
      "long": {
        "generic": "שעון מרכז-מערב אוסטרליה",
        "standard": "שעון מרכז-מערב אוסטרליה (חורף)",
        "daylight": "שעון מרכז-מערב אוסטרליה (קיץ)"
      }
    },
    "Australia_Eastern": {
      "long": {
        "generic": "שעון מזרח אוסטרליה",
        "standard": "שעון מזרח אוסטרליה (חורף)",
        "daylight": "שעון מזרח אוסטרליה (קיץ)"
      }
    },
    "Australia_Western": {
      "long": {
        "generic": "שעון מערב אוסטרליה",
        "standard": "שעון מערב אוסטרליה (חורף)",
        "daylight": "שעון מערב אוסטרליה (קיץ)"
      }
    },
    "Azerbaijan": {
      "long": {
        "generic": "שעון אזרבייג'אן",
        "standard": "שעון אזרבייג'אן (חורף)",
        "daylight": "שעון אזרבייג'אן (קיץ)"
      }
    },
    "Azores": {
      "long": {
        "generic": "שעון האיים האזוריים",
        "standard": "שעון האיים האזוריים (חורף)",
        "daylight": "שעון האיים האזוריים (קיץ)"
      }
    },
    "Bangladesh": {
      "long": {
        "generic": "שעון בנגלדש",
        "standard": "שעון בנגלדש (חורף)",
        "daylight": "שעון בנגלדש (קיץ)"
      }
    },
    "Bhutan": {
      "long": {
        "standard": "שעון בהוטן"
      }
    },
    "Bolivia": {
      "long": {
        "standard": "שעון בוליביה"
      }
    },
    "Brasilia": {
      "long": {
        "generic": "שעון ברזיליה",
        "standard": "שעון ברזיליה (חורף)",

```

```

        "daylight": "שעון ברזיליה (קיץ)"
    },
    },
    "Brunei": {
        "long": {
            "standard": "שעון ברוניי דארוסלאם"
        }
    },
    "Cape_Verde": {
        "long": {
            "generic": "שעון כף ורדה",
            "standard": "שעון כף ורדה (חורף)",
            "daylight": "שעון כף ורדה (קיץ)"
        }
    },
    "Chamorro": {
        "long": {
            "standard": "שעון צ'אמורו"
        }
    },
    "Chatham": {
        "long": {
            "generic": "שעון צ'טהאם",
            "standard": "שעון צ'טהאם (חורף)",
            "daylight": "שעון צ'טהאם (קיץ)"
        }
    },
    "Chile": {
        "long": {
            "generic": "שעון צ'ילה",
            "standard": "שעון צ'ילה (חורף)",
            "daylight": "שעון צ'ילה (קיץ)"
        }
    },
    "China": {
        "long": {
            "generic": "שעון סין",
            "standard": "שעון סין (חורף)",
            "daylight": "שעון סין (קיץ)"
        }
    },
    "Choibalsan": {
        "long": {
            "generic": "שעון צ'ויבלסן",
            "standard": "שעון צ'ויבלסן (חורף)",
            "daylight": "שעון צ'ויבלסן (קיץ)"
        }
    },
    "Christmas": {
        "long": {
            "standard": "שעון האי כריסטמס"
        }
    },
    "Cocos": {
        "long": {
            "standard": "שעון איי קוקוס"
        }
    }

```

```

    },
    "Colombia": {
      "long": {
        "generic": "שעון קולומביה",
        "standard": "שעון קולומביה (חורף)",
        "daylight": "שעון קולומביה (קיץ)"
      }
    },
    "Cook": {
      "long": {
        "generic": "שעון איי קוק",
        "standard": "שעון איי קוק (חורף)",
        "daylight": "שעון איי קוק (מחצית הקיץ)"
      }
    },
    "Cuba": {
      "long": {
        "generic": "שעון קובה",
        "standard": "שעון קובה (חורף)",
        "daylight": "שעון קובה (קיץ)"
      }
    },
    "Davis": {
      "long": {
        "standard": "שעון דייוויס"
      }
    },
    "DumontDUrville": {
      "long": {
        "standard": "שעון דומון ד'אורוויל"
      }
    },
    "East_Timor": {
      "long": {
        "standard": "שעון מזרח טימור"
      }
    },
    "Easter": {
      "long": {
        "generic": "שעון אי הפסחא",
        "standard": "שעון אי הפסחא (חורף)",
        "daylight": "שעון אי הפסחא (קיץ)"
      }
    },
    "Ecuador": {
      "long": {
        "standard": "שעון אקוודור"
      }
    },
    "Europe_Central": {
      "long": {
        "generic": "שעון מרכז אירופה",
        "standard": "שעון מרכז אירופה (חורף)",
        "daylight": "שעון מרכז אירופה (קיץ)"
      }
    },
    "Europe_Eastern": {

```

```

    "long": {
      "generic": "שעון מזרח אירופה",
      "standard": "שעון מזרח אירופה (חורף)",
      "daylight": "שעון מזרח אירופה (קיץ)"
    }
  },
  "Europe_Further_Eastern": {
    "long": {
      "standard": "שעון מינסק"
    }
  },
  "Europe_Western": {
    "long": {
      "generic": "שעון מערב אירופה",
      "standard": "שעון מערב אירופה (חורף)",
      "daylight": "שעון מערב אירופה (קיץ)"
    }
  },
  "Falkland": {
    "long": {
      "generic": "שעון איי פוקלנד",
      "standard": "שעון איי פוקלנד (חורף)",
      "daylight": "שעון איי פוקלנד (קיץ)"
    }
  },
  "Fiji": {
    "long": {
      "generic": "שעון פיג'י",
      "standard": "שעון פיג'י (חורף)",
      "daylight": "שעון פיג'י (קיץ)"
    }
  },
  "French_Guiana": {
    "long": {
      "standard": "שעון גיאנה הצרפתית"
    }
  },
  "French_Southern": {
    "long": {
      "standard": "שעון הארצות הדרומיות והאנטארקטיות של צרפת"
    }
  },
  "Galapagos": {
    "long": {
      "standard": "שעון איי גלאפגוס"
    }
  },
  "Gambier": {
    "long": {
      "standard": "שעון איי גמבייה"
    }
  },
  "Georgia": {
    "long": {
      "generic": "שעון גאורגיה",
      "standard": "שעון גאורגיה (חורף)",
      "daylight": "שעון גאורגיה (קיץ)"
    }
  }
}

```

```

    }
  },
  "Gilbert_Islands": {
    "long": {
      "standard": "שעון איי גילברט"
    }
  },
  "GMT": {
    "long": {
      "standard": "שעון גריניץ'"
    }
  },
  "Greenland_Eastern": {
    "long": {
      "generic": "שעון מזרח גרינלנד",
      "standard": "שעון מזרח גרינלנד (חורף)",
      "daylight": "שעון מזרח גרינלנד (קיץ)"
    }
  },
  "Greenland_Western": {
    "long": {
      "generic": "שעון מערב גרינלנד",
      "standard": "שעון מערב גרינלנד (חורף)",
      "daylight": "שעון מערב גרינלנד (קיץ)"
    }
  },
  "Gulf": {
    "long": {
      "standard": "שעון מדינות המפרץ"
    }
  },
  "Guyana": {
    "long": {
      "standard": "שעון גיאנה"
    }
  },
  "Hawaii_Aleutian": {
    "long": {
      "generic": "שעון האיים האלאוטיים הוואי",
      "standard": "שעון האיים האלאוטיים הוואי (חורף)",
      "daylight": "שעון האיים האלאוטיים הוואי (קיץ)"
    }
  },
  "Hong_Kong": {
    "long": {
      "generic": "שעון הונג קונג",
      "standard": "שעון הונג קונג (חורף)",
      "daylight": "שעון הונג קונג (קיץ)"
    }
  },
  "Hovd": {
    "long": {
      "generic": "שעון חובד",
      "standard": "שעון חובד (חורף)",
      "daylight": "שעון חובד (קיץ)"
    }
  },
},

```



```

"India": {
  "long": {
    "standard": "שעון הודו"
  }
},
"Indian_Ocean": {
  "long": {
    "standard": "שעון האוקיינוס ההודי"
  }
},
"Indochina": {
  "long": {
    "standard": "שעון הודו-סין"
  }
},
"Indonesia_Central": {
  "long": {
    "standard": "שעון מרכז אינדונזיה"
  }
},
"Indonesia_Eastern": {
  "long": {
    "standard": "שעון מזרח אינדונזיה"
  }
},
"Indonesia_Western": {
  "long": {
    "standard": "שעון מערב אינדונזיה"
  }
},
"Iran": {
  "long": {
    "generic": "שעון איראן",
    "standard": "שעון איראן (חורף)",
    "daylight": "שעון איראן (קיץ)"
  }
},
"Irkutsk": {
  "long": {
    "generic": "שעון אירקוטסק",
    "standard": "שעון אירקוטסק (חורף)",
    "daylight": "שעון אירקוטסק (קיץ)"
  }
},
"Israel": {
  "long": {
    "generic": "שעון ישראל",
    "standard": "שעון ישראל (חורף)",
    "daylight": "שעון ישראל (קיץ)"
  }
},
"Japan": {
  "long": {
    "generic": "שעון יפן",
    "standard": "שעון יפן (חורף)",
    "daylight": "שעון יפן (קיץ)"
  }
}

```

```

    },
    "Kamchatka": {
      "long": {
        "generic": "שעון פטרופבלובסק-קמצ'טסקי",
        "standard": "שעון רגיל פטרופבלובסק-קמצ'טסקי",
        "daylight": "שעון קיץ פטרופבלובסק-קמצ'טסקי"
      }
    },
    "Kazakhstan_Eastern": {
      "long": {
        "standard": "שעון מזרח קזחסטן"
      }
    },
    "Kazakhstan_Western": {
      "long": {
        "standard": "שעון מערב קזחסטן"
      }
    },
    "Korea": {
      "long": {
        "generic": "שעון קוריאה",
        "standard": "שעון קוריאה (חורף)",
        "daylight": "שעון קוריאה (קיץ)"
      }
    },
    "Kosrae": {
      "long": {
        "standard": "שעון קוסראה"
      }
    },
    "Krasnoyarsk": {
      "long": {
        "generic": "שעון קרסנויארסק",
        "standard": "שעון קרסנויארסק (חורף)",
        "daylight": "שעון קרסנויארסק (קיץ)"
      }
    },
    "Kyrgystan": {
      "long": {
        "standard": "שעון קירגיזסטן"
      }
    },
    "Line_Islands": {
      "long": {
        "standard": "שעון איי ליין"
      }
    },
    "Lord_Howe": {
      "long": {
        "generic": "שעון אי הלורד האו",
        "standard": "שעון אי הלורד האו (חורף)",
        "daylight": "שעון אי הלורד האו (קיץ)"
      }
    },
    "Macau": {
      "long": {
        "generic": "שעון מקאו",

```

```

        "standard": "שעון חורף מקאו",
        "daylight": "שעון קיץ מקאו"
    },
    },
    "Macquarie": {
        "long": {
            "standard": "שעון מקווארי"
        }
    },
    },
    "Magadan": {
        "long": {
            "generic": "שעון מגדן",
            "standard": "(שעון מגדן) חורף",
            "daylight": "(שעון מגדן) קיץ"
        }
    },
    },
    "Malaysia": {
        "long": {
            "standard": "שעון מלזיה"
        }
    },
    },
    "Maldives": {
        "long": {
            "standard": "שעון האיים המלדיביים"
        }
    },
    },
    "Marquesas": {
        "long": {
            "standard": "שעון איי מרקז"
        }
    },
    },
    "Marshall_Islands": {
        "long": {
            "standard": "שעון איי מרשל"
        }
    },
    },
    "Mauritius": {
        "long": {
            "generic": "שעון מאוריציוס",
            "standard": "(שעון מאוריציוס) חורף",
            "daylight": "(שעון מאוריציוס) קיץ"
        }
    },
    },
    "Mawson": {
        "long": {
            "standard": "שעון מאוסון"
        }
    },
    },
    "Mexico_Northwest": {
        "long": {
            "generic": "שעון צפון-מערב מקסיקו",
            "standard": "(שעון צפון-מערב מקסיקו) חורף",
            "daylight": "(שעון צפון-מערב מקסיקו) קיץ"
        }
    },
    },
    "Mexico_Pacific": {
        "long": {

```

```

        "generic": "שעון מערב מקסיקו",
        "standard": "שעון מערב מקסיקו (חורף)",
        "daylight": "שעון מערב מקסיקו (קיץ)"
    },
},
"Mongolia": {
    "long": {
        "generic": "שעון אולן בטור",
        "standard": "שעון אולן בטור (חורף)",
        "daylight": "שעון אולן בטור (קיץ)"
    }
},
"Moscow": {
    "long": {
        "generic": "שעון מוסקבה",
        "standard": "שעון מוסקבה (חורף)",
        "daylight": "שעון מוסקבה (קיץ)"
    }
},
"Myanmar": {
    "long": {
        "standard": "שעון מיאנמר"
    }
},
"Nauru": {
    "long": {
        "standard": "שעון נאורו"
    }
},
"Nepal": {
    "long": {
        "standard": "שעון נפאל"
    }
},
"New_Caledonia": {
    "long": {
        "generic": "שעון קלדוניה החדשה",
        "standard": "שעון קלדוניה החדשה (חורף)",
        "daylight": "שעון קלדוניה החדשה (קיץ)"
    }
},
"New_Zealand": {
    "long": {
        "generic": "שעון ניו זילנד",
        "standard": "שעון ניו זילנד (חורף)",
        "daylight": "שעון ניו זילנד (קיץ)"
    }
},
"Newfoundland": {
    "long": {
        "generic": "שעון ניופאונדלנד",
        "standard": "שעון ניופאונדלנד (חורף)",
        "daylight": "שעון ניופאונדלנד (קיץ)"
    }
},
"Niue": {
    "long": {

```

```

        "standard": "שעון ניואה"
    },
    },
    "Norfolk": {
        "long": {
            "standard": "שעון האי נורפוק"
        }
    },
    },
    "Noronha": {
        "long": {
            "generic": "שעון פרננדו די נורוניה",
            "standard": "שעון פרננדו די נורוניה (חורף)",
            "daylight": "שעון פרננדו די נורוניה (קיץ)"
        }
    },
    },
    "Novosibirsk": {
        "long": {
            "generic": "שעון נובוסיבירסק",
            "standard": "שעון נובוסיבירסק (חורף)",
            "daylight": "שעון נובוסיבירסק (קיץ)"
        }
    },
    },
    "Omsk": {
        "long": {
            "generic": "שעון אומסק",
            "standard": "שעון אומסק (חורף)",
            "daylight": "שעון אומסק (קיץ)"
        }
    },
    },
    "Pakistan": {
        "long": {
            "generic": "שעון פקיסטן",
            "standard": "שעון פקיסטן (חורף)",
            "daylight": "שעון פקיסטן (קיץ)"
        }
    },
    },
    "Palau": {
        "long": {
            "standard": "שעון פלאו"
        }
    },
    },
    "Papua_New_Guinea": {
        "long": {
            "standard": "שעון פפואה גיניאה החדשה"
        }
    },
    },
    "Paraguay": {
        "long": {
            "generic": "שעון פרגוואי",
            "standard": "שעון פרגוואי (חורף)",
            "daylight": "שעון פרגוואי (קיץ)"
        }
    },
    },
    "Peru": {
        "long": {
            "generic": "שעון פרו",
            "standard": "שעון פרו (חורף)",

```

```

        "daylight": "שעון פרו (קיץ)"
    },
    },
    "Philippines": {
        "long": {
            "generic": "שעון הפיליפינים",
            "standard": "שעון הפיליפינים (חורף)",
            "daylight": "שעון הפיליפינים (קיץ)"
        }
    },
    "Phoenix_Islands": {
        "long": {
            "standard": "שעון איי פיניקס"
        }
    },
    "Pierre_Miquelon": {
        "long": {
            "generic": "שעון סנט פייר ומיקלון",
            "standard": "שעון סנט פייר ומיקלון (חורף)",
            "daylight": "שעון סנט פייר ומיקלון (קיץ)"
        }
    },
    "Pitcairn": {
        "long": {
            "standard": "שעון פיטקרן"
        }
    },
    "Ponape": {
        "long": {
            "standard": "שעון פונאפי"
        }
    },
    "Pyongyang": {
        "long": {
            "standard": "שעון פיונגיאנג"
        }
    },
    "Reunion": {
        "long": {
            "standard": "שעון ראוניון"
        }
    },
    "Rothera": {
        "long": {
            "standard": "שעון רותרה"
        }
    },
    "Sakhalin": {
        "long": {
            "generic": "שעון סחלין",
            "standard": "שעון סחלין (חורף)",
            "daylight": "שעון סחלין (קיץ)"
        }
    },
    "Samara": {
        "long": {
            "generic": "שעון סמרה",

```

```

        "standard": "שעון רגיל סמרה",
        "daylight": "שעון קיץ סמרה"
    },
    },
    "Samoa": {
        "long": {
            "generic": "שעון סמואה",
            "standard": "שעון סמואה (חורף)",
            "daylight": "שעון סמואה (קיץ)"
        }
    },
    },
    "Seychelles": {
        "long": {
            "standard": "שעון איי סיישל"
        }
    },
    },
    "Singapore": {
        "long": {
            "standard": "שעון סינגפור"
        }
    },
    },
    "Solomon": {
        "long": {
            "standard": "שעון איי שלמה"
        }
    },
    },
    "South_Georgia": {
        "long": {
            "standard": "שעון דרום ג'ורג'יה"
        }
    },
    },
    "Suriname": {
        "long": {
            "standard": "שעון סורינאם"
        }
    },
    },
    "Syowa": {
        "long": {
            "standard": "שעון סיווה"
        }
    },
    },
    "Tahiti": {
        "long": {
            "standard": "שעון טהיטי"
        }
    },
    },
    "Taipei": {
        "long": {
            "generic": "שעון טאיפיי",
            "standard": "שעון טאיפיי (חורף)",
            "daylight": "שעון טאיפיי (קיץ)"
        }
    },
    },
    "Tajikistan": {
        "long": {
            "standard": "שעון טג'יקיסטן"
        }
    }
}

```

```

    },
    "Tokelau": {
      "long": {
        "standard": "שעון טוקלאו"
      }
    },
    "Tonga": {
      "long": {
        "generic": "שעון טונגה",
        "standard": "שעון טונגה (חורף)",
        "daylight": "שעון טונגה (קיץ)"
      }
    },
    "Truk": {
      "long": {
        "standard": "שעון צ'וק"
      }
    },
    "Turkmenistan": {
      "long": {
        "generic": "שעון טורקמניסטן",
        "standard": "שעון טורקמניסטן (חורף)",
        "daylight": "שעון טורקמניסטן (קיץ)"
      }
    },
    "Tuvalu": {
      "long": {
        "standard": "שעון טובאלו"
      }
    },
    "Uruguay": {
      "long": {
        "generic": "שעון אורוגוואי",
        "standard": "שעון אורוגוואי (חורף)",
        "daylight": "שעון אורוגוואי (קיץ)"
      }
    },
    "Uzbekistan": {
      "long": {
        "generic": "שעון אוזבקיסטן",
        "standard": "שעון אוזבקיסטן (חורף)",
        "daylight": "שעון אוזבקיסטן (קיץ)"
      }
    },
    "Vanuatu": {
      "long": {
        "generic": "שעון ונואטו",
        "standard": "שעון ונואטו (חורף)",
        "daylight": "שעון ונואטו (קיץ)"
      }
    },
    "Venezuela": {
      "long": {
        "standard": "שעון ונצואלה"
      }
    },
    "Vladivostok": {

```



```

        "long": {
            "generic": "שעון ולדיווסטוק",
            "standard": "שעון ולדיווסטוק (חורף)",
            "daylight": "שעון ולדיווסטוק (קיץ)"
        }
    },
    "Volgograd": {
        "long": {
            "generic": "שעון וולגוגרד",
            "standard": "שעון וולגוגרד (חורף)",
            "daylight": "שעון וולגוגרד (קיץ)"
        }
    },
    "Vostok": {
        "long": {
            "standard": "שעון ווסטוק"
        }
    },
    "Wake": {
        "long": {
            "standard": "שעון האי וייק"
        }
    },
    "Wallis": {
        "long": {
            "standard": "שעון וואליס ופוטונה"
        }
    },
    "Yakutsk": {
        "long": {
            "generic": "שעון יקוטסק",
            "standard": "שעון יקוטסק (חורף)",
            "daylight": "שעון יקוטסק (קיץ)"
        }
    },
    "Yekaterinburg": {
        "long": {
            "generic": "שעון יקטרינבורג",
            "standard": "שעון יקטרינבורג (חורף)",
            "daylight": "שעון יקטרינבורג (קיץ)"
        }
    }
}

```

TIMEZONENAMES.JSX

```

{
    "main";
    {
        "he";
        {

```

```

    "identity";
    {
      "version";
      {
        "_number";
        "$Revision: 13259 $",
        "_cldrVersion";
        "31.0.1";
      }
      "language";
      "he";
    }
    "dates";
    {
      "timeZoneNames";
      {
        "hourFormat";
        "+HH:mm;-HH:mm",
        "gmtFormat";
        "GMT{0}",
        "gmtZeroFormat";
        "GMT",
        "regionFormat";
        "שעול {0}",
        "regionFormat-type-daylight";
        ")ק'י {0 { שעול",
        "regionFormat-type-standard";
        ")ק'י {0 { שעול",
        "fallbackFormat";
        "{1} ({0})",
        "zone";
        {
          "America";
          {
            "Adak";
            {
              "exemplarCity";
              "אדאק";
            }
            "Anchorage";
            {
              "exemplarCity";
              "אנקורג'";
            }
            "Anguilla";
            {
              "exemplarCity";
              "אנגוויילה";
            }
            "Antigua";
            {
              "exemplarCity";
              "אנטיגואה";
            }
            "Araguaina";
            {
              "exemplarCity";

```

```
        "אראגווינה";
    }
    "Argentina";
    {
        "Rio_Gallegos";
        {
            "exemplarCity";
            "ריו גאייגוס";
        }
        "San_Juan";
        {
            "exemplarCity";
            "סן חואן";
        }
        "Ushuaia";
        {
            "exemplarCity";
            "אושואיה";
        }
        "La_Rioja";
        {
            "exemplarCity";
            "לה ריוחה";
        }
        "San_Luis";
        {
            "exemplarCity";
            "סן לואיס";
        }
        "Salta";
        {
            "exemplarCity";
            "סלטה";
        }
        "Tucuman";
        {
            "exemplarCity";
            "טוקומן";
        }
    }
    "Aruba";
    {
        "exemplarCity";
        "ארובה";
    }
    "Asuncion";
    {
        "exemplarCity";
        "אסונסיון";
    }
    "Bahia";
    {
        "exemplarCity";
        "באהיה";
    }
    "Bahia_Banderas";
    {
```

```
        "exemplarCity";
        "באהיה בנדרס";
    }
    "Barbados";
    {
        "exemplarCity";
        "ברבדוס";
    }
    "Belem";
    {
        "exemplarCity";
        "בלם";
    }
    "Belize";
    {
        "exemplarCity";
        "בליז";
    }
    "Blanc-Sablon";
    {
        "exemplarCity";
        "בלאן-סבלון";
    }
    "Boa_Vista";
    {
        "exemplarCity";
        "בואה ויסטה";
    }
    "Bogota";
    {
        "exemplarCity";
        "בוגוטה";
    }
    "Boise";
    {
        "exemplarCity";
        "בוויסי";
    }
    "Buenos_Aires";
    {
        "exemplarCity";
        "בואנוס איירס";
    }
    "Cambridge_Bay";
    {
        "exemplarCity";
        "קיימברידג' ביי";
    }
    "Campo_Grande";
    {
        "exemplarCity";
        "קמפו גרנדה";
    }
    "Cancun";
    {
        "exemplarCity";
        "קנקון";
    }
```

```
}
"Caracas";
{
  "exemplarCity";
  "קראקס";
}
"Catamarca";
{
  "exemplarCity";
  "קטמרקה";
}
"Cayenne";
{
  "exemplarCity";
  "קאיינ";
}
"Cayman";
{
  "exemplarCity";
  "קיימן";
}
"Chicago";
{
  "exemplarCity";
  "שיקגו";
}
"Chihuahua";
{
  "exemplarCity";
  "צ'יוואוואה";
}
"Coral_Harbour";
{
  "exemplarCity";
  "אטיקוקן";
}
"Cordoba";
{
  "exemplarCity";
  "קורדובה";
}
"Costa_Rica";
{
  "exemplarCity";
  "קוסטה ריקה";
}
"Creston";
{
  "exemplarCity";
  "קרטטון";
}
"Cuiaba";
{
  "exemplarCity";
  "קויאבה";
}
"Curacao";
```

```
{
  "exemplarCity";
  "קוראטאו";
}
"Danmarkshavn";
{
  "exemplarCity";
  "דנמרקסהוון";
}
"Dawson";
{
  "exemplarCity";
  "דוסון";
}
"Dawson_Creek";
{
  "exemplarCity";
  "דוסון קריק";
}
"Denver";
{
  "exemplarCity";
  "דנוור";
}
"Detroit";
{
  "exemplarCity";
  "דטרויט";
}
"Dominica";
{
  "exemplarCity";
  "דומיניקה";
}
"Edmonton";
{
  "exemplarCity";
  "אדמונטון";
}
"Eirunepe";
{
  "exemplarCity";
  "אירונפי";
}
"El_Salvador";
{
  "exemplarCity";
  "אל סלבדור";
}
"Fort_Nelson";
{
  "exemplarCity";
  "פורט נלסון";
}
"Fortaleza";
{
  "exemplarCity";
```

```
        "פורטאלזה";
    }
    "Glace_Bay";
    {
        "exemplarCity";
        "גלייט ביי";
    }
    "Godthab";
    {
        "exemplarCity";
        "נואוק";
    }
    "Goose_Bay";
    {
        "exemplarCity";
        "גוסט ביי";
    }
    "Grand_Turk";
    {
        "exemplarCity";
        "גרנד טורק";
    }
    "Grenada";
    {
        "exemplarCity";
        "גרנדה";
    }
    "Guadeloupe";
    {
        "exemplarCity";
        "גואדלופ";
    }
    "Guatemala";
    {
        "exemplarCity";
        "גואטמלה";
    }
    "Guayaquil";
    {
        "exemplarCity";
        "גואיאקיל";
    }
    "Guyana";
    {
        "exemplarCity";
        "גיאנה";
    }
    "Halifax";
    {
        "exemplarCity";
        "הליפקס";
    }
    "Havana";
    {
        "exemplarCity";
        "הוואנה";
    }
}
```

```
"Hermosillo";
{
  "exemplarCity";
  "הרמוסילו";
}
"Indiana";
{
  "Vincennes";
  {
    "exemplarCity";
    "ווינסנס, אינדיאנה";
  }
  "Petersburg";
  {
    "exemplarCity";
    "פיטרסבורג, אינדיאנה";
  }
  "Tell_City";
  {
    "exemplarCity";
    "טל סיטי, אינדיאנה";
  }
  "Knox";
  {
    "exemplarCity";
    "נוקס, אינדיאנה";
  }
  "Winamac";
  {
    "exemplarCity";
    "ווינמאק, אינדיאנה";
  }
  "Marengo";
  {
    "exemplarCity";
    "מרנגו, אינדיאנה";
  }
  "Vevay";
  {
    "exemplarCity";
    "ויוואיי, אינדיאנה";
  }
}
"Indianapolis";
{
  "exemplarCity";
  "אינדיאנפוליס";
}
"Inuvik";
{
  "exemplarCity";
  "אינוויק";
}
"Iqaluit";
{
  "exemplarCity";
  "איגלוויט";
}
```



```
"Jamaica";
{
    "exemplarCity";
    "ג'מייקה";
}
"Jujuy";
{
    "exemplarCity";
    "יוזיו";
}
"Juneau";
{
    "exemplarCity";
    "ג'ונו";
}
"Kentucky";
{
    "Monticello";
    {
        "exemplarCity";
        "מונטיצלו, קנטאקי";
    }
}
"Kralendijk";
{
    "exemplarCity";
    "קרלנדייק";
}
"La_Paz";
{
    "exemplarCity";
    "לאה פאס";
}
"Lima";
{
    "exemplarCity";
    "לימה";
}
"Los_Angeles";
{
    "exemplarCity";
    "לוס אנג'לס";
}
"Louisville";
{
    "exemplarCity";
    "לואיוויג';
}
"Lower_Princes";
{
    "exemplarCity";
    "לואור פרינסס קוורטר";
}
"Maceio";
{
    "exemplarCity";
```

```
        "מסייאו";
    }
    "Managua";
    {
        "exemplarCity";
        "מנגואה";
    }
    "Manaus";
    {
        "exemplarCity";
        "מנאוס";
    }
    "Marigot";
    {
        "exemplarCity";
        "מריגו";
    }
    "Martinique";
    {
        "exemplarCity";
        "מרטיניק";
    }
    "Matamoros";
    {
        "exemplarCity";
        "מטמורוס";
    }
    "Mazatlan";
    {
        "exemplarCity";
        "מזטלן";
    }
    "Mendoza";
    {
        "exemplarCity";
        "מנדוזזה";
    }
    "Menominee";
    {
        "exemplarCity";
        "מנומיני";
    }
    "Merida";
    {
        "exemplarCity";
        "מרידה";
    }
    "Metlakatla";
    {
        "exemplarCity";
        "מטלקטלה";
    }
    "Mexico_City";
    {
        "exemplarCity";
        "מקסיקו סיטי";
    }
}
```

```
"Miquelon";
{
  "exemplarCity";
  "מִיקֶלֹן";
}
"Moncton";
{
  "exemplarCity";
  "מוֹנְקְטוֹן";
}
"Monterrey";
{
  "exemplarCity";
  "מוֹנְטֶרֶי";
}
"Montevideo";
{
  "exemplarCity";
  "מוֹנְטֵוִידֵאוֹ";
}
"Montserrat";
{
  "exemplarCity";
  "מוֹנְטְרַאט";
}
"Nassau";
{
  "exemplarCity";
  "נֶסָאוֹ";
}
"New_York";
{
  "exemplarCity";
  "נְיוֹ יוֹרְק";
}
"Nipigon";
{
  "exemplarCity";
  "נִיפִיגֹן";
}
"Nome";
{
  "exemplarCity";
  "נוֹם";
}
"Noronha";
{
  "exemplarCity";
  "נֹרוֹנְיָה";
}
"North_Dakota";
{
  "Beulah";
  {
    "exemplarCity";
    "בִּיּוּלָה, צֶפוֹן דַּקוֹטָה";
  }
}
```

```
"New_Salem";
{
  "exemplarCity";
  "ניו סיילם, צפון דקוטה";
}
"Center";
{
  "exemplarCity";
  "סנטר, צפון דקוטה";
}
}
"Ojinaga";
{
  "exemplarCity";
  "אוג'ינאגה";
}
"Panama";
{
  "exemplarCity";
  "פנמה";
}
"Pangnirtung";
{
  "exemplarCity";
  "פנגנירטונג";
}
"Paramaribo";
{
  "exemplarCity";
  "פרמריבו";
}
"Phoenix";
{
  "exemplarCity";
  "פיניקס";
}
"Port-au-Prince";
{
  "exemplarCity";
  "פורט או פראנס";
}
"Port_of_Spain";
{
  "exemplarCity";
  "פורט אוף ספייין";
}
"Porto_Velho";
{
  "exemplarCity";
  "פורטו וליו";
}
"Puerto_Rico";
{
  "exemplarCity";
  "פוארטו ריקו";
}
}
"Rainy_River";
```

```
{
  "exemplarCity";
  "רייני ריבר";
}
"Rankin_Inlet";
{
  "exemplarCity";
  "רנקין אינלט";
}
"Recife";
{
  "exemplarCity";
  "רסיפה";
}
"Regina";
{
  "exemplarCity";
  "רג'ינה";
}
"Resolute";
{
  "exemplarCity";
  "רזולוט";
}
"Rio_Branco";
{
  "exemplarCity";
  "ריו ברנקו";
}
"Santa_Isabel";
{
  "exemplarCity";
  "סנטה איסבל";
}
"Santarem";
{
  "exemplarCity";
  "סנטרם";
}
"Santiago";
{
  "exemplarCity";
  "סנטיאגו";
}
"Santo_Domingo";
{
  "exemplarCity";
  "סנטו דומינגו";
}
"Sao_Paulo";
{
  "exemplarCity";
  "סאו פאולו";
}
"Scoresbysund";
{
  "exemplarCity";
```

```
        "סקורטביסונד";
    }
    "Sitka";
    {
        "exemplarCity";
        "סיטקה";
    }
    "St_Barthelemy";
    {
        "exemplarCity";
        "סנט ברתלמי";
    }
    "St_Johns";
    {
        "exemplarCity";
        "סנט ג'ונס";
    }
    "St_Kitts";
    {
        "exemplarCity";
        "סנט קיטס";
    }
    "St_Lucia";
    {
        "exemplarCity";
        "סנט לוסיה";
    }
    "St_Thomas";
    {
        "exemplarCity";
        "סנט תומאס";
    }
    "St_Vincent";
    {
        "exemplarCity";
        "סנט וינסנט";
    }
    "Swift_Current";
    {
        "exemplarCity";
        "סוויפט קרנט";
    }
    "Tegucigalpa";
    {
        "exemplarCity";
        "טגוסיגלפה";
    }
    "Thule";
    {
        "exemplarCity";
        "תולה";
    }
    "Thunder_Bay";
    {
        "exemplarCity";
        "ת'אנדר ביי";
    }
}
```

```
"Tijuana";
{
  "exemplarCity";
  "טיחואנה";
}
"Toronto";
{
  "exemplarCity";
  "טורונטו";
}
"Tortola";
{
  "exemplarCity";
  "טורטולה";
}
"Vancouver";
{
  "exemplarCity";
  "ונקובר";
}
"Whitehorse";
{
  "exemplarCity";
  "ווייטהורס";
}
"Winnipeg";
{
  "exemplarCity";
  "וויניפג";
}
"Yakutat";
{
  "exemplarCity";
  "יקוטאט";
}
"Yellowknife";
{
  "exemplarCity";
  "ילונייף";
}
}
"Atlantic";
{
  "Azores";
  {
    "exemplarCity";
    "האיים האזוריים";
  }
  "Bermuda";
  {
    "exemplarCity";
    "ברמודה";
  }
  "Canary";
  {
    "exemplarCity";
    "האיים הקנריים";
  }
}
```

```
}
"Cape_Verde";
{
  "exemplarCity";
  "כף ורדה";
}
"Faeroe";
{
  "exemplarCity";
  "פארו";
}
"Madeira";
{
  "exemplarCity";
  "מדירה";
}
"Reykjavik";
{
  "exemplarCity";
  "רייקיאויק";
}
"South_Georgia";
{
  "exemplarCity";
  "דרום ג'ורג'יה";
}
"St_Helena";
{
  "exemplarCity";
  "סנט הלנה";
}
"Stanley";
{
  "exemplarCity";
  "סטנלי";
}
}
"Europe";
{
  "Amsterdam";
  {
    "exemplarCity";
    "אמסטרדם";
  }
  "Andorra";
  {
    "exemplarCity";
    "אנדורה";
  }
  "Astrakhan";
  {
    "exemplarCity";
    "אסטרן";
  }
  "Athens";
  {
    "exemplarCity";
```



```

        "אתונה";
    }
    "Belgrade";
    {
        "exemplarCity";
        "בלגרד";
    }
    "Berlin";
    {
        "exemplarCity";
        "ברלין";
    }
    "Bratislava";
    {
        "exemplarCity";
        "ברטיסלבה";
    }
    "Brussels";
    {
        "exemplarCity";
        "בריסל";
    }
    "Bucharest";
    {
        "exemplarCity";
        "בוקרשט";
    }
    "Budapest";
    {
        "exemplarCity";
        "בודפשט";
    }
    "Busingen";
    {
        "exemplarCity";
        "ביזינגן";
    }
    "Chisinau";
    {
        "exemplarCity";
        "קיישינב";
    }
    "Copenhagen";
    {
        "exemplarCity";
        "קופנהגן";
    }
    "Dublin";
    {
        "long";
        {
            "daylight";
            "שעון קיץ אירלנד";
        }
        "exemplarCity";
        "דבלין";
    }
}

```

```
"Gibraltar";
{
  "exemplarCity";
  "גיברלטר";
}
"Guernsey";
{
  "exemplarCity";
  "גרנזי";
}
"Helsinki";
{
  "exemplarCity";
  "הלסינקי";
}
"Isle_of_Man";
{
  "exemplarCity";
  "האי מאן";
}
"Istanbul";
{
  "exemplarCity";
  "איסטנבול";
}
"Jersey";
{
  "exemplarCity";
  "ג'רזי";
}
"Kaliningrad";
{
  "exemplarCity";
  "קלינינגרד";
}
"Kiev";
{
  "exemplarCity";
  "קייב";
}
"Kirov";
{
  "exemplarCity";
  "קירוב";
}
"Lisbon";
{
  "exemplarCity";
  "ליסבון";
}
"Ljubljana";
{
  "exemplarCity";
  "לובליאנה";
}
"London";
{
```

```
        "long";
        {
            "daylight";
            "שעון קיץ בריטניה";
        }
        "exemplarCity";
        "לונדון";
    }
    "Luxembourg";
    {
        "exemplarCity";
        "לוקסמבורג";
    }
    "Madrid";
    {
        "exemplarCity";
        "מדריד";
    }
    "Malta";
    {
        "exemplarCity";
        "מלטה";
    }
    "Mariehamn";
    {
        "exemplarCity";
        "מריהאמן";
    }
    "Minsk";
    {
        "exemplarCity";
        "מינסק";
    }
    "Monaco";
    {
        "exemplarCity";
        "מונקו";
    }
    "Moscow";
    {
        "exemplarCity";
        "מוסקבה";
    }
    "Oslo";
    {
        "exemplarCity";
        "אוסלו";
    }
    "Paris";
    {
        "exemplarCity";
        "פריז";
    }
    "Podgorica";
    {
        "exemplarCity";
        "פודגוריצה";
    }
}
```

```
}
"Prague";
{
  "exemplarCity";
  "פראג";
}
"Riga";
{
  "exemplarCity";
  "ריגה";
}
"Rome";
{
  "exemplarCity";
  "רומא";
}
"Samara";
{
  "exemplarCity";
  "סמרה";
}
"San_Marino";
{
  "exemplarCity";
  "סן מרינו";
}
"Sarajevo";
{
  "exemplarCity";
  "סרייבו";
}
"Simferopol";
{
  "exemplarCity";
  "סימפרופול";
}
"Skopje";
{
  "exemplarCity";
  "סקופיה";
}
"Sofia";
{
  "exemplarCity";
  "סופיה";
}
"Stockholm";
{
  "exemplarCity";
  "שטוקהולם";
}
"Tallinn";
{
  "exemplarCity";
  "טאלין";
}
"Tirane";
```

```
{
  "exemplarCity";
  "טירנה";
}
"Ulyanovsk";
{
  "exemplarCity";
  "אוליאנובסק";
}
"Uzhgorod";
{
  "exemplarCity";
  "אוז'הורוד";
}
"Vaduz";
{
  "exemplarCity";
  "ואדוץ";
}
"Vatican";
{
  "exemplarCity";
  "הוותיקן";
}
"Vienna";
{
  "exemplarCity";
  "וינה";
}
"Vilnius";
{
  "exemplarCity";
  "וילנה";
}
"Volgograd";
{
  "exemplarCity";
  "וולגוגרד";
}
"Warsaw";
{
  "exemplarCity";
  "ורשה";
}
"Zagreb";
{
  "exemplarCity";
  "זאגרב";
}
"Zaporozhye";
{
  "exemplarCity";
  "זפורוז'יה";
}
"Zurich";
{
  "exemplarCity";
```

```
        "ציריך";
    }
}
"Africa";
{
    "Abidjan";
    {
        "exemplarCity";
        "אביג'אן";
    }
    "Accra";
    {
        "exemplarCity";
        "אקרה";
    }
    "Addis_Ababa";
    {
        "exemplarCity";
        "אדיס אבבה";
    }
    "Algiers";
    {
        "exemplarCity";
        "אלג'יר";
    }
    "Asmera";
    {
        "exemplarCity";
        "אסמרה";
    }
    "Bamako";
    {
        "exemplarCity";
        "במאקו";
    }
    "Bangui";
    {
        "exemplarCity";
        "בנגווי";
    }
    "Banjul";
    {
        "exemplarCity";
        "בנג'ול";
    }
    "Bissau";
    {
        "exemplarCity";
        "ביסאו";
    }
    "Blantyre";
    {
        "exemplarCity";
        "בלנטיר";
    }
    "Brazzaville";
    {
```

```
        "exemplarCity";
        "ברזוויל";
    }
    "Bujumbura";
    {
        "exemplarCity";
        "בוג'ומבורה";
    }
    "Cairo";
    {
        "exemplarCity";
        "קהיר";
    }
    "Casablanca";
    {
        "exemplarCity";
        "קזבלנקה";
    }
    "Ceuta";
    {
        "exemplarCity";
        "סאוטה";
    }
    "Conakry";
    {
        "exemplarCity";
        "קונאקרי";
    }
    "Dakar";
    {
        "exemplarCity";
        "דקאר";
    }
    "Dar_es_Salaam";
    {
        "exemplarCity";
        "דאר-א-סלאם";
    }
    "Djibouti";
    {
        "exemplarCity";
        "ג'יבוטי";
    }
    "Douala";
    {
        "exemplarCity";
        "דואלה";
    }
    "El_Aaiun";
    {
        "exemplarCity";
        "אל עיון";
    }
    "Freetown";
    {
        "exemplarCity";
        "פריטאון";
    }
```

```
}
"Gaborone";
{
  "exemplarCity";
  "גבורונה";
}
"Harare";
{
  "exemplarCity";
  "הרארה";
}
"Johannesburg";
{
  "exemplarCity";
  "יוהנסבורג";
}
"Juba";
{
  "exemplarCity";
  "ג'ובה";
}
"Kampala";
{
  "exemplarCity";
  "קמפאלה";
}
"Khartoum";
{
  "exemplarCity";
  "חרטום";
}
"Kigali";
{
  "exemplarCity";
  "קיגלי";
}
"Kinshasa";
{
  "exemplarCity";
  "קינשסה";
}
"Lagos";
{
  "exemplarCity";
  "לגוס";
}
"Libreville";
{
  "exemplarCity";
  "ליברוויל";
}
"Lome";
{
  "exemplarCity";
  "לומה";
}
"Luanda";
```



```
{
  "exemplarCity";
  "לואנדה";
}
"Lubumbashi";
{
  "exemplarCity";
  "לובומבאשי";
}
"Lusaka";
{
  "exemplarCity";
  "לוטסקה";
}
"Malabo";
{
  "exemplarCity";
  "מלבו";
}
"Maputo";
{
  "exemplarCity";
  "מאפוטו";
}
"Maseru";
{
  "exemplarCity";
  "מסרו";
}
"Mbabane";
{
  "exemplarCity";
  "אמבאבאנה";
}
"Mogadishu";
{
  "exemplarCity";
  "מוגדישו";
}
"Monrovia";
{
  "exemplarCity";
  "מונרוביה";
}
"Nairobi";
{
  "exemplarCity";
  "ניירובי";
}
"Ndjamena";
{
  "exemplarCity";
  "נג'מנה";
}
"Niamey";
{
  "exemplarCity";
```

```

        "ניאמיי";
    }
    "Nouakchott";
    {
        "exemplarCity";
        "נואקצ'וט";
    }
    "Ouagadougou";
    {
        "exemplarCity";
        "וואגאדוגו";
    }
    "Porto-Novo";
    {
        "exemplarCity";
        "פורטו נובו";
    }
    "Sao_Tome";
    {
        "exemplarCity";
        "סאו טומה";
    }
    "Tripoli";
    {
        "exemplarCity";
        "טריפולי";
    }
    "Tunis";
    {
        "exemplarCity";
        "תוניס";
    }
    "Windhoek";
    {
        "exemplarCity";
        "ווינדהוק";
    }
}
"Asia";
{
    "Aden";
    {
        "exemplarCity";
        "עדן";
    }
    "Almaty";
    {
        "exemplarCity";
        "אלמאטי";
    }
    "Amman";
    {
        "exemplarCity";
        "עמאן";
    }
    "Anadyr";
    {

```

```
        "exemplarCity";
        "אנדיר";
    }
    "Aqtau";
    {
        "exemplarCity";
        "אקטאו";
    }
    "Aqtobe";
    {
        "exemplarCity";
        "אקטובה";
    }
    "Ashgabat";
    {
        "exemplarCity";
        "אשגבט";
    }
    "Baghdad";
    {
        "exemplarCity";
        "בגדד";
    }
    "Bahrain";
    {
        "exemplarCity";
        "בחריין";
    }
    "Baku";
    {
        "exemplarCity";
        "באקו";
    }
    "Bangkok";
    {
        "exemplarCity";
        "בנגקוק";
    }
    "Barnaul";
    {
        "exemplarCity";
        "ברנאול";
    }
    "Beirut";
    {
        "exemplarCity";
        "ביירות";
    }
    "Bishkek";
    {
        "exemplarCity";
        "בישקק";
    }
    "Brunei";
    {
        "exemplarCity";
        "ברוניי";
    }
}
```

```
}  
"Calcutta";  
{  
  "exemplarCity";  
  "קולקטטה";  
}  
"Chita";  
{  
  "exemplarCity";  
  "צ'יטה";  
}  
"Choibalsan";  
{  
  "exemplarCity";  
  "צ'ויבלסן";  
}  
"Colombo";  
{  
  "exemplarCity";  
  "קולומבו";  
}  
"Damascus";  
{  
  "exemplarCity";  
  "דמשק";  
}  
"Dhaka";  
{  
  "exemplarCity";  
  "דאקה";  
}  
"Dili";  
{  
  "exemplarCity";  
  "דילי";  
}  
"Dubai";  
{  
  "exemplarCity";  
  "דובאי";  
}  
"Dushanbe";  
{  
  "exemplarCity";  
  "דושנבה";  
}  
"Gaza";  
{  
  "exemplarCity";  
  "עזה";  
}  
"Hebron";  
{  
  "exemplarCity";  
  "חברון";  
}  
"Hong_Kong";
```

```
{
  "exemplarCity";
  "הונג קונג";
}
"Hovd";
{
  "exemplarCity";
  "חובד";
}
"Irkutsk";
{
  "exemplarCity";
  "אירקוטסק";
}
"Jakarta";
{
  "exemplarCity";
  "ג'קרטה";
}
"Jayapura";
{
  "exemplarCity";
  "ג'איאפורה";
}
"Jerusalem";
{
  "exemplarCity";
  "ירושלים";
}
"Kabul";
{
  "exemplarCity";
  "קאבול";
}
"Kamchatka";
{
  "exemplarCity";
  "קמצ'טקה";
}
"Karachi";
{
  "exemplarCity";
  "קראצ'י";
}
"Katmandu";
{
  "exemplarCity";
  "קטמנדו";
}
"Khandyga";
{
  "exemplarCity";
  "חנדיגה";
}
"Krasnoyarsk";
{
  "exemplarCity";
```

```
        "קרסנויארסק";
    }
    "Kuala_Lumpur";
    {
        "exemplarCity";
        "קואלה לומפור";
    }
    "Kuching";
    {
        "exemplarCity";
        "קוצ'ינג";
    }
    "Kuwait";
    {
        "exemplarCity";
        "כווית";
    }
    "Macau";
    {
        "exemplarCity";
        "מקאו";
    }
    "Magadan";
    {
        "exemplarCity";
        "מגדל";
    }
    "Makassar";
    {
        "exemplarCity";
        "מאקאסאר";
    }
    "Manila";
    {
        "exemplarCity";
        "מנילה";
    }
    "Muscat";
    {
        "exemplarCity";
        "מוסקט";
    }
    "Nicosia";
    {
        "exemplarCity";
        "ניקוסיה";
    }
    "Novokuznetsk";
    {
        "exemplarCity";
        "נובוקוזנטסק";
    }
    "Novosibirsk";
    {
        "exemplarCity";
        "נובוסירסק";
    }
}
```

```
"Omsk";
{
  "exemplarCity";
  "אומסק";
}
"Oral";
{
  "exemplarCity";
  "אורל";
}
"Phnom_Penh";
{
  "exemplarCity";
  "פנום פן";
}
"Pontianak";
{
  "exemplarCity";
  "פונטיאנק";
}
"Pyongyang";
{
  "exemplarCity";
  "פיונגיאנג";
}
"Qatar";
{
  "exemplarCity";
  "קטאר";
}
"Qyzylorda";
{
  "exemplarCity";
  "קזילורדה";
}
"Rangoon";
{
  "exemplarCity";
  "רנגון";
}
"Riyadh";
{
  "exemplarCity";
  "ריאד";
}
"Saigon";
{
  "exemplarCity";
  "הו צ'י מין סיטי";
}
"Sakhalin";
{
  "exemplarCity";
  "סחלין";
}
"Samarkand";
{
```

```
        "exemplarCity";
        "סמרקנד";
    }
    "Seoul";
    {
        "exemplarCity";
        "סיאול";
    }
    "Shanghai";
    {
        "exemplarCity";
        "שנחאי";
    }
    "Singapore";
    {
        "exemplarCity";
        "סינגפור";
    }
    "Srednekolymsk";
    {
        "exemplarCity";
        "סרדנליקולימסק";
    }
    "Taipei";
    {
        "exemplarCity";
        "טאייפיי";
    }
    "Tashkent";
    {
        "exemplarCity";
        "טשקנט";
    }
    "Tbilisi";
    {
        "exemplarCity";
        "טביליסי";
    }
    "Tehran";
    {
        "exemplarCity";
        "טהרן";
    }
    "Thimphu";
    {
        "exemplarCity";
        "טהימפהו";
    }
    "Tokyo";
    {
        "exemplarCity";
        "טוקיו";
    }
    "Tomsk";
    {
        "exemplarCity";
        "טומסק";
    }
}
```



```

    }
    "Ulaanbaatar";
    {
        "exemplarCity";
        "אולאאנבטאר";
    }
    "Urumqi";
    {
        "exemplarCity";
        "אורומקי";
    }
    "Ust-Nera";
    {
        "exemplarCity";
        "אוסט-נרה";
    }
    "Vientiane";
    {
        "exemplarCity";
        "האנוי";
    }
    "Vladivostok";
    {
        "exemplarCity";
        "ולדייוסטוק";
    }
    "Yakutsk";
    {
        "exemplarCity";
        "יקוטסק";
    }
    "Yekaterinburg";
    {
        "exemplarCity";
        "יקטרינבורג";
    }
    "Yerevan";
    {
        "exemplarCity";
        "ירואן";
    }
    }
    "Indian";
    {
        "Antananarivo";
        {
            "exemplarCity";
            "אנטננריבו";
        }
        "Chagos";
        {
            "exemplarCity";
            "צ'אגוס";
        }
        "Christmas";
        {
            "exemplarCity";

```

```

        "האי כריסטמס";
    }
    "Cocos";
    {
        "exemplarCity";
        "קוקוס";
    }
    "Comoro";
    {
        "exemplarCity";
        "קומורו";
    }
    "Kerguelen";
    {
        "exemplarCity";
        "קרגוולן";
    }
    "Mahe";
    {
        "exemplarCity";
        "מהא";
    }
    "Maldives";
    {
        "exemplarCity";
        "האיים המלדיביים";
    }
    "Mauritius";
    {
        "exemplarCity";
        "מאוריציוס";
    }
    "Mayotte";
    {
        "exemplarCity";
        "מאיוט";
    }
    "Reunion";
    {
        "exemplarCity";
        "ראוניון";
    }
}
"Australia";
{
    "Adelaide";
    {
        "exemplarCity";
        "אדלייד";
    }
    "Brisbane";
    {
        "exemplarCity";
        "בריסביילן";
    }
    "Broken_Hill";
    {

```

```
        "exemplarCity";
        "ברוקן היל";
    }
    "Currie";
    {
        "exemplarCity";
        "קרִי";
    }
    "Darwin";
    {
        "exemplarCity";
        "דרווין";
    }
    "Eucla";
    {
        "exemplarCity";
        "יוקלה";
    }
    "Hobart";
    {
        "exemplarCity";
        "הוברט";
    }
    "Lindeman";
    {
        "exemplarCity";
        "לינדמן";
    }
    "Lord_Howe";
    {
        "exemplarCity";
        "אִי הֶלֹרֵד הָאוּ";
    }
    "Melbourne";
    {
        "exemplarCity";
        "מלבורן";
    }
    "Perth";
    {
        "exemplarCity";
        "פֶּרְתִּי";
    }
    "Sydney";
    {
        "exemplarCity";
        "סִידְנִי";
    }
}
"Pacific";
{
    "Apia";
    {
        "exemplarCity";
        "אפיה";
    }
    "Auckland";
```

```
{
  "exemplarCity";
  "אוקלנד";
}
"Bougainville";
{
  "exemplarCity";
  "בוגנוויל";
}
"Chatham";
{
  "exemplarCity";
  "צ'אטהאם";
}
"Easter";
{
  "exemplarCity";
  "איי הפסחא";
}
"Efate";
{
  "exemplarCity";
  "אפטטה";
}
"Enderbury";
{
  "exemplarCity";
  "אנדרבורי";
}
"Fakaofo";
{
  "exemplarCity";
  "פקאופו";
}
"Fiji";
{
  "exemplarCity";
  "פיג'י";
}
"Funafuti";
{
  "exemplarCity";
  "פונפוט";
}
"Galapagos";
{
  "exemplarCity";
  "גלפאגוס";
}
"Gambier";
{
  "exemplarCity";
  "איי גמבייה";
}
"Guadalcanal";
{
  "exemplarCity";
```

```

        "גוודלוקנאל";
    }
    "Guam";
    {
        "exemplarCity";
        "גואם";
    }
    "Honolulu";
    {
        "exemplarCity";
        "הונוולולו";
    }
    "Johnston";
    {
        "exemplarCity";
        "ג'ונסטון";
    }
    "Kiritimati";
    {
        "exemplarCity";
        "קיריטימאטי";
    }
    "Kosrae";
    {
        "exemplarCity";
        "קוסרה";
    }
    "Kwajalein";
    {
        "exemplarCity";
        "קוואג'ליין";
    }
    "Majuro";
    {
        "exemplarCity";
        "מאג'ורו";
    }
    "Marquesas";
    {
        "exemplarCity";
        "מארקייז";
    }
    "Midway";
    {
        "exemplarCity";
        "מידוויי";
    }
    "Nauru";
    {
        "exemplarCity";
        "נאורו";
    }
    "Niue";
    {
        "exemplarCity";
        "ניואה";
    }
}

```

```
"Norfolk";
{
  "exemplarCity";
  "נורפוק";
}
"Noumea";
{
  "exemplarCity";
  "נומאה";
}
"Pago_Pago";
{
  "exemplarCity";
  "פאגו פאגו";
}
"Palau";
{
  "exemplarCity";
  "פלאו";
}
"Pitcairn";
{
  "exemplarCity";
  "פיטקרן";
}
"Ponape";
{
  "exemplarCity";
  "פונפיי";
}
"Port_Moresby";
{
  "exemplarCity";
  "פורט מורסבי";
}
"Rarotonga";
{
  "exemplarCity";
  "רארוטונגה";
}
"Saipan";
{
  "exemplarCity";
  "סאיפאן";
}
"Tahiti";
{
  "exemplarCity";
  "טהיטי";
}
"Tarawa";
{
  "exemplarCity";
  "טאראווה";
}
"Tongatapu";
{
```

```

        "exemplarCity";
        "טונגטאפּו";
    }
    "Truk";
    {
        "exemplarCity";
        "צ'וק";
    }
    "Wake";
    {
        "exemplarCity";
        "קייק";
    }
    "Wallis";
    {
        "exemplarCity";
        "ווליס";
    }
}
"Arctic";
{
    "Longyearbyen";
    {
        "exemplarCity";
        "לונגיירביון";
    }
}
"Antarctica";
{
    "Casey";
    {
        "exemplarCity";
        "קאסיי";
    }
    "Davis";
    {
        "exemplarCity";
        "דייוויס";
    }
    "DumontDUrville";
    {
        "exemplarCity";
        "דומון ד'אורוויל";
    }
    "Macquarie";
    {
        "exemplarCity";
        "מקרי";
    }
    "Mawson";
    {
        "exemplarCity";
        "מוסון";
    }
    "McMurdo";
    {
        "exemplarCity";
    }
}

```

```

        "מק-מרדו";
    }
    "Palmer";
    {
        "exemplarCity";
        "פאלמר";
    }
    "Rothera";
    {
        "exemplarCity";
        "רות'רה";
    }
    "Syowa";
    {
        "exemplarCity";
        "סיוואה";
    }
    "Troll";
    {
        "exemplarCity";
        "טרול";
    }
    "Vostok";
    {
        "exemplarCity";
        "ווסטוק";
    }
}
"Etc";
{
    "GMT";
    {
        "exemplarCity";
        "GMT";
    }
    "GMT1";
    {
        "exemplarCity";
        "GMT+1";
    }
    "GMT10";
    {
        "exemplarCity";
        "GMT+10";
    }
    "GMT11";
    {
        "exemplarCity";
        "GMT+11";
    }
    "GMT12";
    {
        "exemplarCity";
        "GMT+12";
    }
    "GMT2";
    {

```



```
        "exemplarCity";
        "GMT+2";
    }
    "GMT3";
    {
        "exemplarCity";
        "GMT+3";
    }
    "GMT4";
    {
        "exemplarCity";
        "GMT+4";
    }
    "GMT5";
    {
        "exemplarCity";
        "GMT+5";
    }
    "GMT6";
    {
        "exemplarCity";
        "GMT+6";
    }
    "GMT7";
    {
        "exemplarCity";
        "GMT+7";
    }
    "GMT8";
    {
        "exemplarCity";
        "GMT+8";
    }
    "GMT9";
    {
        "exemplarCity";
        "GMT+9";
    }
    "GMT-1";
    {
        "exemplarCity";
        "GMT-1";
    }
    "GMT-10";
    {
        "exemplarCity";
        "GMT-10";
    }
    "GMT-11";
    {
        "exemplarCity";
        "GMT-11";
    }
    "GMT-12";
    {
        "exemplarCity";
        "GMT-12";
    }
```

```
}
"GMT-13";
{
  "exemplarCity";
  "GMT-13";
}
"GMT-14";
{
  "exemplarCity";
  "GMT-14";
}
"GMT-2";
{
  "exemplarCity";
  "GMT-2";
}
"GMT-3";
{
  "exemplarCity";
  "GMT-3";
}
"GMT-4";
{
  "exemplarCity";
  "GMT-4";
}
"GMT-5";
{
  "exemplarCity";
  "GMT-5";
}
"GMT-6";
{
  "exemplarCity";
  "GMT-6";
}
"GMT-7";
{
  "exemplarCity";
  "GMT-7";
}
"GMT-8";
{
  "exemplarCity";
  "GMT-8";
}
"GMT-9";
{
  "exemplarCity";
  "GMT-9";
}
"UTC";
{
  "long";
  {
    "standard";
    "זמן אוניברסלי מתואם";
  }
}
```

```

    }
    "short";
    {
      "standard";
      "UTC";
    }
    "exemplarCity";
    "UTC";
  }
  "Unknown";
  {
    "exemplarCity";
    "עיר לא ידועה";
  }
}
"metazone";
{
  "Afghanistan";
  {
    "long";
    {
      "standard";
      "שעון אפגניסטן";
    }
  }
  "Africa_Central";
  {
    "long";
    {
      "standard";
      "שעון מרכז אפריקה";
    }
  }
  "Africa_Eastern";
  {
    "long";
    {
      "standard";
      "שעון מזרח אפריקה";
    }
  }
  "Africa_Southern";
  {
    "long";
    {
      "standard";
      "שעון דרום אפריקה";
    }
  }
  "Africa_Western";
  {
    "long";
    {
      "generic";
      "שעון מערב אפריקה",
      "standard";
    }
  }
}

```

```

        "שעון מערב אפריקה (חורף)",
        "daylight";
        "שעון מערב אפריקה (קיץ)";
    }
}
"Alaska";
{
    "long";
    {
        "generic";
        "שעון אלסקה",
        "standard";
        "שעון אלסקה (חורף)",
        "daylight";
        "שעון אלסקה (קיץ)";
    }
}
"Amazon";
{
    "long";
    {
        "generic";
        "שעון אמזונס",
        "standard";
        "שעון אמזונס (חורף)",
        "daylight";
        "שעון אמזונס (קיץ)";
    }
}
"America_Central";
{
    "long";
    {
        "generic";
        "שעון מרכז ארה"ב",
        "standard";
        "שעון מרכז ארה"ב (חורף)",
        "daylight";
        "שעון מרכז ארה"ב (קיץ)";
    }
}
"America_Eastern";
{
    "long";
    {
        "generic";
        "שעון החוף המזרחי",
        "standard";
        "שעון החוף המזרחי (חורף)",
        "daylight";
        "שעון החוף המזרחי (קיץ)";
    }
}
"America_Mountain";
{
    "long";
    {

```

```

        "generic";
        "שעון אזור ההרים בארה"ב",
        "standard";
        "שעון אזור ההרים בארה"ב (חורף)",
        "daylight";
        "שעון אזור ההרים בארה"ב (קיץ)";
    }
}
"America_Pacific";
{
    "long";
    {
        "generic";
        "שעון מערב ארה"ב",
        "standard";
        "שעון מערב ארה"ב (חורף)",
        "daylight";
        "שעון מערב ארה"ב (קיץ)";
    }
}
"Anadyr";
{
    "long";
    {
        "generic";
        "שעון אנדיר",
        "standard";
        "שעון רגיל אנדיר",
        "daylight";
        "שעון קיץ אנדיר";
    }
}
"Apia";
{
    "long";
    {
        "generic";
        "שעון אפיה",
        "standard";
        "שעון אפיה (חורף)",
        "daylight";
        "שעון אפיה (קיץ)";
    }
}
"Arabian";
{
    "long";
    {
        "generic";
        "שעון חצי האי ערב",
        "standard";
        "שעון חצי האי ערב (חורף)",
        "daylight";
        "שעון חצי האי ערב (קיץ)";
    }
}
"Argentina";

```

```

    {
      "long";
      {
        "generic";
        "שעון ארגנטינה",
        "standard";
        "שעון ארגנטינה (חורף)",
        "daylight";
        "שעון ארגנטינה (קיץ)";
      }
    }
    "Argentina_Western";
    {
      "long";
      {
        "generic";
        "שעון מערב ארגנטינה",
        "standard";
        "שעון מערב ארגנטינה (חורף)",
        "daylight";
        "שעון מערב ארגנטינה (קיץ)";
      }
    }
    "Armenia";
    {
      "long";
      {
        "generic";
        "שעון ארמניה",
        "standard";
        "שעון ארמניה (חורף)",
        "daylight";
        "שעון ארמניה (קיץ)";
      }
    }
    "Atlantic";
    {
      "long";
      {
        "generic";
        "שעון האוקיינוס האטלנטי",
        "standard";
        "שעון האוקיינוס האטלנטי (חורף)",
        "daylight";
        "שעון האוקיינוס האטלנטי (קיץ)";
      }
    }
    "Australia_Central";
    {
      "long";
      {
        "generic";
        "שעון מרכז אוסטרליה",
        "standard";
        "שעון מרכז אוסטרליה (חורף)",
        "daylight";
        "שעון מרכז אוסטרליה (קיץ)";
      }
    }
  }

```

```
    }  
  }  
  "Australia_CentralWestern";  
  {  
    "long";  
    {  
      "generic";  
      "שעון מרכז-מערב אוסטרליה",  
      "standard";  
      "שעון מרכז-מערב אוסטרליה (חורף)",  
      "daylight";  
      "שעון מרכז-מערב אוסטרליה (קיץ)";  
    }  
  }  
  "Australia_Eastern";  
  {  
    "long";  
    {  
      "generic";  
      "שעון מזרח אוסטרליה",  
      "standard";  
      "שעון מזרח אוסטרליה (חורף)",  
      "daylight";  
      "שעון מזרח אוסטרליה (קיץ)";  
    }  
  }  
  "Australia_Western";  
  {  
    "long";  
    {  
      "generic";  
      "שעון מערב אוסטרליה",  
      "standard";  
      "שעון מערב אוסטרליה (חורף)",  
      "daylight";  
      "שעון מערב אוסטרליה (קיץ)";  
    }  
  }  
  "Azerbaijan";  
  {  
    "long";  
    {  
      "generic";  
      "שעון אזרבייג'אן",  
      "standard";  
      "שעון אזרבייג'אן (חורף)",  
      "daylight";  
      "שעון אזרבייג'אן (קיץ)";  
    }  
  }  
  "Azores";  
  {  
    "long";  
    {  
      "generic";  
      "שעון האיים האזוריים",  
      "standard";  
    }  
  }
```

```
        "שעון האיים האזוריים (חורף)",
        "daylight";
        "שעון האיים האזוריים (קיץ)";
    }
}
"Bangladesh";
{
    "long";
    {
        "generic";
        "שעון בנגלדש",
        "standard";
        "שעון בנגלדש (חורף)",
        "daylight";
        "שעון בנגלדש (קיץ)";
    }
}
"Bhutan";
{
    "long";
    {
        "standard";
        "שעון בהוטן";
    }
}
"Bolivia";
{
    "long";
    {
        "standard";
        "שעון בוליביה";
    }
}
"Brasilia";
{
    "long";
    {
        "generic";
        "שעון ברזיליה",
        "standard";
        "שעון ברזיליה (חורף)",
        "daylight";
        "שעון ברזיליה (קיץ)";
    }
}
"Brunei";
{
    "long";
    {
        "standard";
        "שעון ברוניי דארוסלאם";
    }
}
"Cape_Verde";
{
    "long";
    {
```



```
        "generic";
        "שעון כף ורדה",
        "standard";
        "שעון כף ורדה) חורף",
        "daylight";
        "שעון כף ורדה) קיץ";
    }
}
"Chamorro";
{
    "long";
    {
        "standard";
        "שעון צ'אמורו";
    }
}
"Chatham";
{
    "long";
    {
        "generic";
        "שעון צ'טהאם",
        "standard";
        "שעון צ'טהאם) חורף",
        "daylight";
        "שעון צ'טהאם) קיץ";
    }
}
"Chile";
{
    "long";
    {
        "generic";
        "שעון צ'ילה",
        "standard";
        "שעון צ'ילה) חורף",
        "daylight";
        "שעון צ'ילה) קיץ";
    }
}
"China";
{
    "long";
    {
        "generic";
        "שעון סין",
        "standard";
        "שעון סין) חורף",
        "daylight";
        "שעון סין) קיץ";
    }
}
"Choibalsan";
{
    "long";
    {
        "generic";
```

```
        "שעון צ'ויבלסן",
        "standard";
        "שעון צ'ויבלסן (חורף)",
        "daylight";
        "שעון צ'ויבלסן (קיץ)";
    }
}
"Christmas";
{
    "long";
    {
        "standard";
        "שעון האי כריסטמס";
    }
}
"Cocos";
{
    "long";
    {
        "standard";
        "שעון איי קוקוס";
    }
}
"Colombia";
{
    "long";
    {
        "generic";
        "שעון קולומביה",
        "standard";
        "שעון קולומביה (חורף)",
        "daylight";
        "שעון קולומביה (קיץ)";
    }
}
"Cook";
{
    "long";
    {
        "generic";
        "שעון איי קוק",
        "standard";
        "שעון איי קוק (חורף)",
        "daylight";
        "שעון איי קוק (מחצית הקיץ)";
    }
}
"Cuba";
{
    "long";
    {
        "generic";
        "שעון קובה",
        "standard";
        "שעון קובה (חורף)",
        "daylight";
        "שעון קובה (קיץ)";
    }
}
```

```
    }  
  }  
  "Davis";  
  {  
    "long";  
    {  
      "standard";  
      "שעון דיוויס";  
    }  
  }  
  "DumontDUrville";  
  {  
    "long";  
    {  
      "standard";  
      "שעון דומון ד'אורוויל";  
    }  
  }  
  "East_Timor";  
  {  
    "long";  
    {  
      "standard";  
      "שעון מזרח טימור";  
    }  
  }  
  "Easter";  
  {  
    "long";  
    {  
      "generic";  
      "שעון אי הפסחא",  
      "standard";  
      "שעון אי הפסחא (חורף)",  
      "daylight";  
      "שעון אי הפסחא (קיץ)";  
    }  
  }  
  "Ecuador";  
  {  
    "long";  
    {  
      "standard";  
      "שעון אקוודור";  
    }  
  }  
  "Europe_Central";  
  {  
    "long";  
    {  
      "generic";  
      "שעון מרכז אירופה",  
      "standard";  
      "שעון מרכז אירופה (חורף)",  
      "daylight";  
      "שעון מרכז אירופה (קיץ)";  
    }  
  }
```

```

    }
    "Europe_Eastern";
    {
      "long";
      {
        "generic";
        "שעון מזרח אירופה",
        "standard";
        "שעון מזרח אירופה (חורף)",
        "daylight";
        "שעון מזרח אירופה (קיץ)";
      }
    }
    "Europe_Further_Eastern";
    {
      "long";
      {
        "standard";
        "שעון מינסק";
      }
    }
    "Europe_Western";
    {
      "long";
      {
        "generic";
        "שעון מערב אירופה",
        "standard";
        "שעון מערב אירופה (חורף)",
        "daylight";
        "שעון מערב אירופה (קיץ)";
      }
    }
    "Falkland";
    {
      "long";
      {
        "generic";
        "שעון איי פוקלנד",
        "standard";
        "שעון איי פוקלנד (חורף)",
        "daylight";
        "שעון איי פוקלנד (קיץ)";
      }
    }
    "Fiji";
    {
      "long";
      {
        "generic";
        "שעון פיג'י",
        "standard";
        "שעון פיג'י (חורף)",
        "daylight";
        "שעון פיג'י (קיץ)";
      }
    }
  }

```

```
"French_Guiana";
{
  "long";
  {
    "standard";
    "שעון גיאנה הצרפתית";
  }
}
"French_Southern";
{
  "long";
  {
    "standard";
    "שעון הארצות הדרומיות והאנטארקטיות של צרפת";
  }
}
"Galapagos";
{
  "long";
  {
    "standard";
    "שעון איי גלאפגוס";
  }
}
"Gambier";
{
  "long";
  {
    "standard";
    "שעון איי גמבייה";
  }
}
"Georgia";
{
  "long";
  {
    "generic";
    "שעון גאורגיה",
    "standard";
    "שעון גאורגיה (חורף)",
    "daylight";
    "שעון גאורגיה (קיץ)";
  }
}
"Gilbert_Islands";
{
  "long";
  {
    "standard";
    "שעון איי גילברט";
  }
}
"GMT";
{
  "long";
  {
    "standard";
```

```

        "שעון גריניץ'";
    }
}
"Greenland_Eastern";
{
    "long";
    {
        "generic";
        "שעון מזרח גרינלנד",
        "standard";
        "שעון מזרח גרינלנד (חורף)",
        "daylight";
        "שעון מזרח גרינלנד (קיץ)";
    }
}
"Greenland_Western";
{
    "long";
    {
        "generic";
        "שעון מערב גרינלנד",
        "standard";
        "שעון מערב גרינלנד (חורף)",
        "daylight";
        "שעון מערב גרינלנד (קיץ)";
    }
}
"Gulf";
{
    "long";
    {
        "standard";
        "שעון מדינות המפרץ";
    }
}
"Guyana";
{
    "long";
    {
        "standard";
        "שעון גיאנה";
    }
}
"Hawaii_Aleutian";
{
    "long";
    {
        "generic";
        "שעון האיים האלאוטיים הוואי",
        "standard";
        "שעון האיים האלאוטיים הוואי (חורף)",
        "daylight";
        "שעון האיים האלאוטיים הוואי (קיץ)";
    }
}
"Hong_Kong";
{

```

```
        "long";
        {
            "generic";
            "שעון הונג קונג",
            "standard";
            "שעון הונג קונג (חורף)",
            "daylight";
            "שעון הונג קונג (קיץ)";
        }
    }
    "Hovd";
    {
        "long";
        {
            "generic";
            "שעון חובד",
            "standard";
            "שעון חובד (חורף)",
            "daylight";
            "שעון חובד (קיץ)";
        }
    }
    "India";
    {
        "long";
        {
            "standard";
            "שעון הודו";
        }
    }
    "Indian_Ocean";
    {
        "long";
        {
            "standard";
            "שעון האוקיינוס ההודי";
        }
    }
    "Indochina";
    {
        "long";
        {
            "standard";
            "שעון הודו-סין";
        }
    }
    "Indonesia_Central";
    {
        "long";
        {
            "standard";
            "שעון מרכז אינדונזיה";
        }
    }
    "Indonesia_Eastern";
    {
        "long";
```

```

        {
            "standard";
            "שעון מזרח אינדונזיה";
        }
    }
    "Indonesia_Western";
    {
        "long";
        {
            "standard";
            "שעון מערב אינדונזיה";
        }
    }
    "Iran";
    {
        "long";
        {
            "generic";
            "שעון איראן",
            "standard";
            "שעון איראן (חורף)",
            "daylight";
            "שעון איראן (קיץ)";
        }
    }
    "Irkutsk";
    {
        "long";
        {
            "generic";
            "שעון אירקוטסק",
            "standard";
            "שעון אירקוטסק (חורף)",
            "daylight";
            "שעון אירקוטסק (קיץ)";
        }
    }
    "Israel";
    {
        "long";
        {
            "generic";
            "שעון ישראל",
            "standard";
            "שעון ישראל (חורף)",
            "daylight";
            "שעון ישראל (קיץ)";
        }
    }
    "Japan";
    {
        "long";
        {
            "generic";
            "שעון יפן",
            "standard";
            "שעון יפן (חורף)",

```



```

        "daylight";
        "שעון יפן (קיץ)";
    }
}
"Kamchatka";
{
    "long";
    {
        "generic";
        "שעון פטרופלובסק-קמצ'טסקי",
        "standard";
        "שעון רגיל פטרופלובסק-קמצ'טסקי",
        "daylight";
        "שעון קיץ פטרופלובסק-קמצ'טסקי";
    }
}
"Kazakhstan_Eastern";
{
    "long";
    {
        "standard";
        "שעון מזרח קזחסטן";
    }
}
"Kazakhstan_Western";
{
    "long";
    {
        "standard";
        "שעון מערב קזחסטן";
    }
}
"Korea";
{
    "long";
    {
        "generic";
        "שעון קוריאה",
        "standard";
        "שעון קוריאה (חורף)",
        "daylight";
        "שעון קוריאה (קיץ)";
    }
}
"Kosrae";
{
    "long";
    {
        "standard";
        "שעון קוסראה";
    }
}
"Krasnoyarsk";
{
    "long";
    {
        "generic";

```

```

        "שעון קרסנויארסק",
        "standard";
        "שעון קרסנויארסק (חורף)",
        "daylight";
        "שעון קרסנויארסק (קיץ)";
    }
}
"Kyrgystan";
{
    "long";
    {
        "standard";
        "שעון קירגיזסטן";
    }
}
"Line_Islands";
{
    "long";
    {
        "standard";
        "שעון איי ליין";
    }
}
"Lord_Howe";
{
    "long";
    {
        "generic";
        "שעון אי הלורד האו",
        "standard";
        "שעון אי הלורד האו (חורף)",
        "daylight";
        "שעון אי הלורד האו (קיץ)";
    }
}
"Macau";
{
    "long";
    {
        "generic";
        "שעון מקאו",
        "standard";
        "שעון חורף מקאו",
        "daylight";
        "שעון קיץ מקאו";
    }
}
"Macquarie";
{
    "long";
    {
        "standard";
        "שעון מקווארי";
    }
}
"Magadan";
{

```

```

        "long";
        {
            "generic";
            "שעון מגדן",
            "standard";
            "שעון מגדן (חורף)",
            "daylight";
            "שעון מגדן (קיץ)";
        }
    }
    "Malaysia";
    {
        "long";
        {
            "standard";
            "שעון מלזיה";
        }
    }
    "Maldives";
    {
        "long";
        {
            "standard";
            "שעון האיים המלדיביים";
        }
    }
    "Marquesas";
    {
        "long";
        {
            "standard";
            "שעון איי מרקז";
        }
    }
    "Marshall_Islands";
    {
        "long";
        {
            "standard";
            "שעון איי מרשל";
        }
    }
    "Mauritius";
    {
        "long";
        {
            "generic";
            "שעון מאוריציוס",
            "standard";
            "שעון מאוריציוס (חורף)",
            "daylight";
            "שעון מאוריציוס (קיץ)";
        }
    }
    "Mawson";
    {
        "long";
    }

```

```

        {
            "standard";
            "שעון מאוסון";
        }
    }
    "Mexico_Northwest";
    {
        "long";
        {
            "generic";
            "שעון צפון-מערב מקסיקו",
            "standard";
            "שעון צפון-מערב מקסיקו (חורף)",
            "daylight";
            "שעון צפון-מערב מקסיקו (קיץ)";
        }
    }
    "Mexico_Pacific";
    {
        "long";
        {
            "generic";
            "שעון מערב מקסיקו",
            "standard";
            "שעון מערב מקסיקו (חורף)",
            "daylight";
            "שעון מערב מקסיקו (קיץ)";
        }
    }
    "Mongolia";
    {
        "long";
        {
            "generic";
            "שעון אולן בטור",
            "standard";
            "שעון אולן בטור (חורף)",
            "daylight";
            "שעון אולן בטור (קיץ)";
        }
    }
    "Moscow";
    {
        "long";
        {
            "generic";
            "שעון מוסקבה",
            "standard";
            "שעון מוסקבה (חורף)",
            "daylight";
            "שעון מוסקבה (קיץ)";
        }
    }
    "Myanmar";
    {
        "long";
        {

```

```
        "standard";
        "שעון מיאנמר";
    }
}
"Nauru";
{
    "long";
    {
        "standard";
        "שעון נאורו";
    }
}
"Nepal";
{
    "long";
    {
        "standard";
        "שעון נפאל";
    }
}
"New_Caledonia";
{
    "long";
    {
        "generic";
        "שעון קלדוניה החדשה",
        "standard";
        "שעון קלדוניה החדשה (חורף)",
        "daylight";
        "שעון קלדוניה החדשה (קיץ)";
    }
}
"New_Zealand";
{
    "long";
    {
        "generic";
        "שעון ניו זילנד",
        "standard";
        "שעון ניו זילנד (חורף)",
        "daylight";
        "שעון ניו זילנד (קיץ)";
    }
}
"Newfoundland";
{
    "long";
    {
        "generic";
        "שעון ניופאונדלנד",
        "standard";
        "שעון ניופאונדלנד (חורף)",
        "daylight";
        "שעון ניופאונדלנד (קיץ)";
    }
}
"Niue";
```

```

    {
      "long";
      {
        "standard";
        "שעון ניואה";
      }
    }
    "Norfolk";
    {
      "long";
      {
        "standard";
        "שעון האי נורפוק";
      }
    }
    "Noronha";
    {
      "long";
      {
        "generic";
        "שעון פרננדו די נורוניה",
        "standard";
        "שעון פרננדו די נורוניה (חורף)",
        "daylight";
        "שעון פרננדו די נורוניה (קיץ)";
      }
    }
    "Novosibirsk";
    {
      "long";
      {
        "generic";
        "שעון נובוסיבירסק",
        "standard";
        "שעון נובוסיבירסק (חורף)",
        "daylight";
        "שעון נובוסיבירסק (קיץ)";
      }
    }
    "Omsk";
    {
      "long";
      {
        "generic";
        "שעון אומסק",
        "standard";
        "שעון אומסק (חורף)",
        "daylight";
        "שעון אומסק (קיץ)";
      }
    }
    "Pakistan";
    {
      "long";
      {
        "generic";
        "שעון פקיסטן",

```

```

        "standard";
        "שעון פקיסטן (חורף)",
        "daylight";
        "שעון פקיסטן (קיץ)";
    }
}
"Palau";
{
    "long";
    {
        "standard";
        "שעון פלאו";
    }
}
"Papua_New_Guinea";
{
    "long";
    {
        "standard";
        "שעון פפואה גיניאה החדשה";
    }
}
"Paraguay";
{
    "long";
    {
        "generic";
        "שעון פרגוואי",
        "standard";
        "שעון פרגוואי (חורף)",
        "daylight";
        "שעון פרגוואי (קיץ)";
    }
}
"Peru";
{
    "long";
    {
        "generic";
        "שעון פרו",
        "standard";
        "שעון פרו (חורף)",
        "daylight";
        "שעון פרו (קיץ)";
    }
}
"Philippines";
{
    "long";
    {
        "generic";
        "שעון הפיליפינים",
        "standard";
        "שעון הפיליפינים (חורף)",
        "daylight";
        "שעון הפיליפינים (קיץ)";
    }
}

```

```
}
"Phoenix_Islands";
{
  "long";
  {
    "standard";
    "שעון איי פֿיניקס";
  }
}
"Pierre_Miquelon";
{
  "long";
  {
    "generic";
    "שעון סנט פייר ומיקלון",
    "standard";
    "שעון סנט פייר ומיקלון (חורף)",
    "daylight";
    "שעון סנט פייר ומיקלון (קיץ)";
  }
}
"Pitcairn";
{
  "long";
  {
    "standard";
    "שעון פיטקרן";
  }
}
"Ponape";
{
  "long";
  {
    "standard";
    "שעון פֿונאפֿי";
  }
}
"Pyongyang";
{
  "long";
  {
    "standard";
    "שעון פֿיונגיאנג";
  }
}
"Reunion";
{
  "long";
  {
    "standard";
    "שעון ראוניון";
  }
}
"Rothera";
{
  "long";
  {
```



```

        "standard";
        "שעון רות'רה";
    }
}
"Sakhalin";
{
    "long";
    {
        "generic";
        "שעון סחלין",
        "standard";
        "שעון סחלין (חורף)",
        "daylight";
        "שעון סחלין (קיץ)";
    }
}
"Samara";
{
    "long";
    {
        "generic";
        "שעון סמרה",
        "standard";
        "שעון רגיל סמרה",
        "daylight";
        "שעון קיץ סמרה";
    }
}
"Samoa";
{
    "long";
    {
        "generic";
        "שעון סמואה",
        "standard";
        "שעון סמואה (חורף)",
        "daylight";
        "שעון סמואה (קיץ)";
    }
}
"Seychelles";
{
    "long";
    {
        "standard";
        "שעון איי סיישל";
    }
}
"Singapore";
{
    "long";
    {
        "standard";
        "שעון סינגפור";
    }
}
"Solomon";

```

```
{
  "long";
  {
    "standard";
    "שעון איי שלמה";
  }
}
"South_Georgia";
{
  "long";
  {
    "standard";
    "שעון דרום ג'ורג'יה";
  }
}
"Suriname";
{
  "long";
  {
    "standard";
    "שעון סורינאם";
  }
}
"Syowa";
{
  "long";
  {
    "standard";
    "שעון סייווה";
  }
}
"Tahiti";
{
  "long";
  {
    "standard";
    "שעון טהיטי";
  }
}
"Taipei";
{
  "long";
  {
    "generic";
    "שעון טאיפיי",
    "standard";
    "שעון טאיפיי (חורף)",
    "daylight";
    "שעון טאיפיי (קיץ)";
  }
}
"Tajikistan";
{
  "long";
  {
    "standard";
    "שעון טג'יקיסטן";
  }
}
```

```
    }  
  }  
  "Tokelau";  
  {  
    "long";  
    {  
      "standard";  
      "שעון טוקלאו";  
    }  
  }  
  "Tonga";  
  {  
    "long";  
    {  
      "generic";  
      "שעון טונגה",  
      "standard";  
      "(שעון טונגה) חורף",  
      "daylight";  
      "(שעון טונגה) קיץ";  
    }  
  }  
  "Truk";  
  {  
    "long";  
    {  
      "standard";  
      "שעון צ'וק";  
    }  
  }  
  "Turkmenistan";  
  {  
    "long";  
    {  
      "generic";  
      "שעון טורקמניסטן",  
      "standard";  
      "(שעון טורקמניסטן) חורף",  
      "daylight";  
      "(שעון טורקמניסטן) קיץ";  
    }  
  }  
  "Tuvalu";  
  {  
    "long";  
    {  
      "standard";  
      "שעון טובאלו";  
    }  
  }  
  "Uruguay";  
  {  
    "long";  
    {  
      "generic";  
      "שעון אורוגוואי",  
      "standard";  
    }  
  }  
}
```

```

        "שעון אורוגוואי (חורף)",
        "daylight";
        "שעון אורוגוואי (קיץ)";
    }
}
"Uzbekistan";
{
    "long";
    {
        "generic";
        "שעון אוזבקיסטן",
        "standard";
        "שעון אוזבקיסטן (חורף)",
        "daylight";
        "שעון אוזבקיסטן (קיץ)";
    }
}
"Vanuatu";
{
    "long";
    {
        "generic";
        "שעון ונואטו",
        "standard";
        "שעון ונואטו (חורף)",
        "daylight";
        "שעון ונואטו (קיץ)";
    }
}
"Venezuela";
{
    "long";
    {
        "standard";
        "שעון ונצואלה";
    }
}
"Vladivostok";
{
    "long";
    {
        "generic";
        "שעון ולדיווסטוק",
        "standard";
        "שעון ולדיווסטוק (חורף)",
        "daylight";
        "שעון ולדיווסטוק (קיץ)";
    }
}
"Volgograd";
{
    "long";
    {
        "generic";
        "שעון וולגוגרד",
        "standard";
        "שעון וולגוגרד (חורף)",

```

```

        "daylight";
    }
}
"Vostok";
{
    "long";
    {
        "standard";
        "שעון ווסטוק";
    }
}
"Wake";
{
    "long";
    {
        "standard";
        "שעון האי וייק";
    }
}
"Wallis";
{
    "long";
    {
        "standard";
        "שעון וואליס ופוטונה";
    }
}
"Yakutsk";
{
    "long";
    {
        "generic";
        "שעון יקוטסק",
        "standard";
        "שעון יקוטסק) חורף)",
        "daylight";
        "שעון יקוטסק) קיץ)";
    }
}
"Yekaterinburg";
{
    "long";
    {
        "generic";
        "שעון יקטרינבורג",
        "standard";
        "שעון יקטרינבורג) חורף)",
        "daylight";
        "שעון יקטרינבורג) קיץ)";
    }
}
}
}
}
}
}
}
}
}
}

```

```
}
```

[Functional-component]

CA-GREGORIAN.JSON

```
{
  "main": {
    "he": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31.0.1"
        },
        "language": "he"
      },
      "dates": {
        "calendars": {
          "gregorian": {
            "months": {
              "format": {
                "abbreviated": {
                  "1": "ינו",
                  "2": "פבר",
                  "3": "מרץ",
                  "4": "אפר",
                  "5": "מאי",
                  "6": "יוני",
                  "7": "יולי",
                  "8": "אוג",
                  "9": "ספט",
                  "10": "אוק",
                  "11": "נוב",
                  "12": "דצמ"
                },
                "narrow": {
                  "1": "1",
                  "2": "2",
                  "3": "3",
                  "4": "4",
                  "5": "5",
                  "6": "6",
                  "7": "7",
                  "8": "8",
                  "9": "9",
                  "10": "10",
                  "11": "11",
                  "12": "12"
                },
                "wide": {
                  "1": "ינואר",
                  "2": "פברואר",
                  "3": "מרץ",
                  "4": "אפריל",
                  "5": "מאי",
                  "6": "יוני",

```

```

        "7": "יולי",
        "8": "אוגוסט",
        "9": "ספטמבר",
        "10": "אוקטובר",
        "11": "נובמבר",
        "12": "דצמבר"
      }
    },
    "stand-alone": {
      "abbreviated": {
        "1": "ינו'",
        "2": "פבר'",
        "3": "מרץ",
        "4": "אפר'",
        "5": "מאי",
        "6": "יוני",
        "7": "יולי",
        "8": "אוג'",
        "9": "ספט'",
        "10": "אוק'",
        "11": "נוב'",
        "12": "דצמ'"
      },
      "narrow": {
        "1": "1",
        "2": "2",
        "3": "3",
        "4": "4",
        "5": "5",
        "6": "6",
        "7": "7",
        "8": "8",
        "9": "9",
        "10": "10",
        "11": "11",
        "12": "12"
      },
      "wide": {
        "1": "ינואר",
        "2": "פברואר",
        "3": "מרץ",
        "4": "אפריל",
        "5": "מאי",
        "6": "יוני",
        "7": "יולי",
        "8": "אוגוסט",
        "9": "ספטמבר",
        "10": "אוקטובר",
        "11": "נובמבר",
        "12": "דצמבר"
      }
    }
  },
  "days": {
    "format": {
      "abbreviated": {
        "sun": "יום א'",

```

```

        "mon": "יום ב'",
        "tue": "יום ג'",
        "wed": "יום ד'",
        "thu": "יום ה'",
        "fri": "יום ו'",
        "sat": "שבת"
    },
    "narrow": {
        "sun": "א'",
        "mon": "ב'",
        "tue": "ג'",
        "wed": "ד'",
        "thu": "ה'",
        "fri": "ו'",
        "sat": "ש"
    },
    "short": {
        "sun": "א",
        "mon": "ב",
        "tue": "ג",
        "wed": "ד",
        "thu": "ה",
        "fri": "ו",
        "sat": "ש"
    },
    "wide": {
        "sun": "יום ראשון",
        "mon": "יום שני",
        "tue": "יום שלישי",
        "wed": "יום רביעי",
        "thu": "יום חמישי",
        "fri": "יום שישי",
        "sat": "יום שבת"
    }
},
"stand-alone": {
    "abbreviated": {
        "sun": "יום א'",
        "mon": "יום ב'",
        "tue": "יום ג'",
        "wed": "יום ד'",
        "thu": "יום ה'",
        "fri": "יום ו'",
        "sat": "שבת"
    },
    "narrow": {
        "sun": "א'",
        "mon": "ב'",
        "tue": "ג'",
        "wed": "ד'",
        "thu": "ה'",
        "fri": "ו'",
        "sat": "ש"
    },
    "short": {
        "sun": "א",
        "mon": "ב",

```



```

        "tue": "י'ג",
        "wed": "י'ד",
        "thu": "י'ה",
        "fri": "י'ו",
        "sat": "י'ש"
    },
    "wide": {
        "sun": "יום ראשון",
        "mon": "יום שני",
        "tue": "יום שלישי",
        "wed": "יום רביעי",
        "thu": "יום חמישי",
        "fri": "יום שישי",
        "sat": "יום שבת"
    }
},
"quarters": {
    "format": {
        "abbreviated": {
            "1": "Q1",
            "2": "Q2",
            "3": "Q3",
            "4": "Q4"
        },
        "narrow": {
            "1": "1",
            "2": "2",
            "3": "3",
            "4": "4"
        },
        "wide": {
            "1": "1 רבעון",
            "2": "2 רבעון",
            "3": "3 רבעון",
            "4": "4 רבעון"
        }
    },
    "stand-alone": {
        "abbreviated": {
            "1": "Q1",
            "2": "Q2",
            "3": "Q3",
            "4": "Q4"
        },
        "narrow": {
            "1": "1",
            "2": "2",
            "3": "3",
            "4": "4"
        },
        "wide": {
            "1": "1 רבעון",
            "2": "2 רבעון",
            "3": "3 רבעון",
            "4": "4 רבעון"
        }
    }
}

```

```

    },
    "dayPeriods": {
      "format": {
        "abbreviated": {
          "midnight": "חצות",
          "am": "לפנה"צ",
          "pm": "אחה"צ",
          "morning1": "בוקר",
          "afternoon1": "צהריים",
          "afternoon2": "אחר הצהריים",
          "evening1": "ערב",
          "night1": "לילה",
          "night2": "לפנות בוקר"
        },
        "narrow": {
          "midnight": "חצות",
          "am": "לפנה"צ",
          "pm": "אחה"צ",
          "morning1": "בוקר",
          "afternoon1": "צהריים",
          "afternoon2": "אחר הצהריים",
          "evening1": "ערב",
          "night1": "לילה",
          "night2": "לפנות בוקר"
        },
        "wide": {
          "midnight": "חצות",
          "am": "לפנה"צ",
          "pm": "אחה"צ",
          "morning1": "בוקר",
          "afternoon1": "צהריים",
          "afternoon2": "אחר הצהריים",
          "evening1": "ערב",
          "night1": "לילה",
          "night2": "לפנות בוקר"
        }
      },
      "stand-alone": {
        "abbreviated": {
          "midnight": "חצות",
          "am": "לפנה"צ",
          "pm": "אחה"צ",
          "morning1": "בוקר",
          "afternoon1": "צהריים",
          "afternoon2": "אחר הצהריים",
          "evening1": "ערב",
          "night1": "לילה",
          "night2": "לפנות בוקר"
        },
        "narrow": {
          "midnight": "חצות",
          "am": "לפנה"צ",
          "pm": "אחה"צ",
          "morning1": "בוקר",
          "afternoon1": "צהריים",
          "afternoon2": "אחר הצהריים",

```

```

        "evening1": "ערב",
        "night1": "לילה",
        "night2": "לפנות בוקר"
    },
    "wide": {
        "midnight": "חצות",
        "am": "לפנה"צ",
        "pm": "אחה"צ",
        "morning1": "בוקר",
        "afternoon1": "צהריים",
        "afternoon2": "אחר הצהריים",
        "evening1": "ערב",
        "night1": "לילה",
        "night2": "לפנות בוקר"
    }
},
    "eras": {
        "eraNames": {
            "0": "לפני הספירה",
            "0-alt-variant": "לפנה"ס",
            "1": "לספירה",
            "1-alt-variant": "CE"
        },
        "eraAbbr": {
            "0": "לפנה"ס",
            "0-alt-variant": "BCE",
            "1": "לספירה",
            "1-alt-variant": "CE"
        },
        "eraNarrow": {
            "0": "לפנה"ס",
            "0-alt-variant": "BCE",
            "1": "לספירה",
            "1-alt-variant": "CE"
        }
    },
    "dateFormats": {
        "full": "EEEE, d MMMM y",
        "long": "d MMMM y",
        "medium": "d MMM y",
        "short": "d.M.y"
    },
    "timeFormats": {
        "full": "H:mm:ss zzzz",
        "long": "H:mm:ss z",
        "medium": "H:mm:ss",
        "short": "H:mm"
    },
    "dateTimeFormats": {
        "full": "{1} בשעה {0}",
        "long": "{1} בשעה {0}",
        "medium": "{1}, {0}",
        "short": "{1}, {0}",
        "availableFormats": {
            "d": "d",
            "E": "ccc",

```

```

"Ed": "E ה-d",
"Ehm": "E h:mm a",
"EHm": "E H:mm",
"Ehms": "E h:mm:ss a",
"EHms": "E H:mm:ss",
"Gy": "y G",
"GyMMM": "MMM y G",
"GyMMMd": "d מMMM y G",
"GyMMMED": "E, d מMMM y G",
"h": "h a",
"H": "H",
"hm": "h:mm a",
"Hm": "H:mm",
"hms": "h:mm:ss a",
"Hms": "H:mm:ss",
"hmsv": "h:mm:ss a v",
"Hmsv": "HH:mm:ss v",
"hm v": "h:mm a v",
"Hm v": "HH:mm v",
"M": "L",
"Md": "d.M",
"MEd": "E, d.M",
"MMM": "LLL",
"MMMd": "d מMMM",
"MMMED": "E, d מMMM",
"MMMMd": "d מMMMM",
"MMMMW-count-one": "שבוע W מMMM",
"MMMMW-count-two": "שבוע W מMMM",
"MMMMW-count-many": "שבוע W מMMM",
"MMMMW-count-other": "שבוע W מMMM",
"ms": "mm:ss",
"y": "y",
"yM": "M.y",
"yMd": "d.M.y",
"yMEd": "E, d.M.y",
"yMM": "M.y",
"yMMM": "MMM y",
"yMMMd": "d מMMM y",
"yMMMED": "E, d מMMM y",
"yMMMM": "MMMM y",
"yQQQ": "QQQ y",
"yQQQQ": "QQQQ y",
"yw-count-one": "שבוע w בשנת y",
"yw-count-two": "שבוע w בשנת y",
"yw-count-many": "שבוע w בשנת y",
"yw-count-other": "שבוע w בשנת y"
},
"appendItems": {
  "Day": "{0} ({2}: {1})",
  "Day-Of-Week": "{0} {1}",
  "Era": "{1} {0}",
  "Hour": "{0} ({2}: {1})",
  "Minute": "{0} ({2}: {1})",
  "Month": "{0} ({2}: {1})",
  "Quarter": "{0} ({2}: {1})",
  "Second": "{0} ({2}: {1})",
  "Timezone": "{0} {1}",

```

```

    "Week": "{0} ({2}: {1})",
    "Year": "{1} {0}"
  },
  "intervalFormats": {
    "intervalFormatFallback": "{0} - {1}",
    "d": {
      "d": "d-d"
    },
    "h": {
      "a": "h a - h a",
      "h": "h-h a"
    },
    "H": {
      "H": "H-H"
    },
    "hm": {
      "a": "h:mm a - h:mm a",
      "h": "h:mm-h:mm a",
      "m": "h:mm-h:mm a"
    },
    "Hm": {
      "H": "H:mm-H:mm",
      "m": "H:mm-H:mm"
    },
    "hmv": {
      "a": "h:mm a - h:mm a v",
      "h": "h:mm-h:mm a v",
      "m": "h:mm-h:mm a v"
    },
    "Hmv": {
      "H": "H:mm-H:mm v",
      "m": "H:mm-H:mm v"
    },
    "hv": {
      "a": "h a - h a v",
      "h": "h-h a v"
    },
    "Hv": {
      "H": "H-H v"
    },
    "M": {
      "M": "M-M"
    },
    "Md": {
      "d": "d.M-d.M",
      "M": "d.M-d.M"
    },
    "MEd": {
      "d": "EEEE d.M-EEEE d.M",
      "M": "EEEE d.M - EEEE d.M"
    },
    "MMM": {
      "M": "MMM-MMM"
    },
    "MMMd": {
      "d": "d-d  MMM",
      "M": "d  MMM-d  MMM"
    }
  }

```

```

    },
    "MMMed": {
      "d": "EEEE, d 1MMM - EEEE, d 1MMM",
      "M": "EEEE, d 1MMM - EEEE, d 1MMM"
    },
    "MMMM": {
      "M": "LLLL-LLLL"
    },
    "Y": {
      "y": "y-y"
    },
    "yM": {
      "M": "M.y-M.y",
      "y": "M.y-M.y"
    },
    "yMd": {
      "d": "dd.M.y - dd.M.y",
      "M": "d.M.y - d.M.y",
      "y": "d.M.y - d.M.y"
    },
    "yMED": {
      "d": "EEEE d.M.y - EEEE d.M.y",
      "M": "EEEE d.M.y - EEEE d.M.y",
      "y": "EEEE d.M.y - EEEE d.M.y"
    },
    "yMMM": {
      "M": "MMM-MMM y",
      "y": "MMM y - MMM y"
    },
    "yMMMd": {
      "d": "d-d 1MMM y",
      "M": "d MMM - d MMM y",
      "y": "d MMM y - d MMM y"
    },
    "yMMMED": {
      "d": "EEEE d MMM - EEEE d MMM y",
      "M": "EEEE d MMM - EEEE d MMM y",
      "y": "EEEE d MMM y - EEEE d MMM y"
    },
    "yMMMM": {
      "M": "MMMM-MMMM y",
      "y": "MMMM y-MMMM y"
    }
  }
}

```

CA-GREGORIAN.JSX

```
{
    "main";
```

```

{
  "he";
  {
    "identity";
    {
      "version";
      {
        "_number";
        "$Revision: 13259 $",
        "_cldrVersion";
        "31.0.1";
      }
      "language";
      "he";
    }
    "dates";
    {
      "calendars";
      {
        "gregorian";
        {
          "months";
          {
            "format";
            {
              "abbreviated";
              {
                "1";
                "ינו'",
                "2";
                "פבר'",
                "3";
                "מרץ",
                "4";
                "אפר'",
                "5";
                "מאי",
                "6";
                "יוני",
                "7";
                "יולי",
                "8";
                "אוג'",
                "9";
                "ספט'",
                "10";
                "אוק'",
                "11";
                "נוב'",
                "12";
                "דצמ'";
              }
            }
            "narrow";
            {
              "1";
              "1",
              "2";
            }
          }
        }
      }
    }
  }
}

```

```

        "2",
        "3";
    "3",
    "4";
    "4",
    "5";
    "5",
    "6";
    "6",
    "7";
    "7",
    "8";
    "8",
    "9";
    "9",
    "10";
    "10",
    "11";
    "11",
    "12";
    "12";
    }
    "wide";
    {
        "1";
        "ינואר",
        "2";
        "פברואר",
        "3";
        "מרץ",
        "4";
        "אפריל",
        "5";
        "מאי",
        "6";
        "יוני",
        "7";
        "יולי",
        "8";
        "אוגוסט",
        "9";
        "ספטמבר",
        "10";
        "אוקטובר",
        "11";
        "נובמבר",
        "12";
        "דצמבר";
    }
    }
    "stand-alone";
    {
        "abbreviated";
        {
            "1";
            "ינו'",
            "2";

```



```
        "פבר'י",  
        "3";  
        "מרץ",  
        "4";  
        "אפר'י",  
        "5";  
        "מאי",  
        "6";  
        "יוני",  
        "7";  
        "יולי",  
        "8";  
        "אוג'",  
        "9";  
        "ספט'",  
        "10";  
        "אוק'",  
        "11";  
        "נוב'",  
        "12";  
        "דצמ'";  
    }  
    "narrow";  
    {  
        "1";  
        "1",  
        "2";  
        "2",  
        "3";  
        "3",  
        "4";  
        "4",  
        "5";  
        "5",  
        "6";  
        "6",  
        "7";  
        "7",  
        "8";  
        "8",  
        "9";  
        "9",  
        "10";  
        "10",  
        "11";  
        "11",  
        "12";  
        "12";  
    }  
    "wide";  
    {  
        "1";  
        "ינואר",  
        "2";  
        "פברואר",  
        "3";  
        "מרץ",
```

```

        "4";
        "אפריל",
        "5";
        "מאי",
        "6";
        "יוני",
        "7";
        "יולי",
        "8";
        "אוגוסט",
        "9";
        "ספטמבר",
        "10";
        "אוקטובר",
        "11";
        "נובמבר",
        "12";
        "דצמבר";
    }
}
}
"days";
{
    "format";
    {
        "abbreviated";
        {
            "sun";
            "יום א'",
            "mon";
            "יום ב'",
            "tue";
            "יום ג'",
            "wed";
            "יום ד'",
            "thu";
            "יום ה'",
            "fri";
            "יום ו'",
            "sat";
            "שבת";
        }
        "narrow";
        {
            "sun";
            "א'",
            "mon";
            "ב'",
            "tue";
            "ג'",
            "wed";
            "ד'",
            "thu";
            "ה'",
            "fri";
            "ו'",
            "sat";
        }
    }
}

```

```

        "ש'";
    }
    "short";
    {
        "sun";
        "א'",
        "mon";
        "ב'",
        "tue";
        "ג'",
        "wed";
        "ד'",
        "thu";
        "ה'",
        "fri";
        "ו'",
        "sat";
        "ש'";
    }
    "wide";
    {
        "sun";
        "יום ראשון",
        "mon";
        "יום שני",
        "tue";
        "יום שלישי",
        "wed";
        "יום רביעי",
        "thu";
        "יום חמישי",
        "fri";
        "יום שישי",
        "sat";
        "יום שבת";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "sun";
        "יום א'",
        "mon";
        "יום ב'",
        "tue";
        "יום ג'",
        "wed";
        "יום ד'",
        "thu";
        "יום ה'",
        "fri";
        "יום ו'",
        "sat";
        "שבת";
    }
}
"narrow";

```

```

        {
            "sun";
            "א'",
            "mon";
            "ב'",
            "tue";
            "ג'",
            "wed";
            "ד'",
            "thu";
            "ה'",
            "fri";
            "ו'",
            "sat";
            "ש'";
        }
        "short";
        {
            "sun";
            "א'",
            "mon";
            "ב'",
            "tue";
            "ג'",
            "wed";
            "ד'",
            "thu";
            "ה'",
            "fri";
            "ו'",
            "sat";
            "ש'";
        }
        "wide";
        {
            "sun";
            "יום ראשון",
            "mon";
            "יום שני",
            "tue";
            "יום שלישי",
            "wed";
            "יום רביעי",
            "thu";
            "יום חמישי",
            "fri";
            "יום שישי",
            "sat";
            "יום שבת";
        }
    }
    "quarters";
    {
        "format";
        {
            "abbreviated";

```

```

    {
      "1";
      "Q1",
        "2";
      "Q2",
        "3";
      "Q3",
        "4";
      "Q4";
    }
    "narrow";
    {
      "1";
      "1",
        "2";
      "2",
        "3";
      "3",
        "4";
      "4";
    }
    "wide";
    {
      "1";
      "1 רבעון",
        "2";
      "2 רבעון",
        "3";
      "3 רבעון",
        "4";
      "4 רבעון";
    }
  }
  "stand-alone";
  {
    "abbreviated";
    {
      "1";
      "Q1",
        "2";
      "Q2",
        "3";
      "Q3",
        "4";
      "Q4";
    }
    "narrow";
    {
      "1";
      "1",
        "2";
      "2",
        "3";
      "3",
        "4";
      "4";
    }
  }
}

```

```

        "wide";
        {
            "1";
            "1 רבעון",
            "2";
            "2 רבעון",
            "3";
            "3 רבעון",
            "4";
            "4 רבעון";
        }
    }
}
"dayPeriods";
{
    "format";
    {
        "abbreviated";
        {
            "midnight";
            "חצות",
            "am";
            "לפנה"צ",
            "pm";
            "אחה"צ",
            "morning1";
            "בוקר",
            "afternoon1";
            "צהריים",
            "afternoon2";
            "אחר הצהריים",
            "evening1";
            "ערב",
            "night1";
            "לילה",
            "night2";
            "לפנות בוקר";
        }
    }
    "narrow";
    {
        "midnight";
        "חצות",
        "am";
        "לפנה"צ",
        "pm";
        "אחה"צ",
        "morning1";
        "בוקר",
        "afternoon1";
        "צהריים",
        "afternoon2";
        "אחר הצהריים",
        "evening1";
        "ערב",
        "night1";
        "לילה",
        "night2";
    }
}

```

```

        "לפנות בוקר";
    }
    "wide";
    {
        "midnight";
        "חצות",
        "am";
        "לפנה"צ",
        "pm";
        "אחה"צ",
        "morning1";
        "בוקר",
        "afternoon1";
        "צהריים",
        "afternoon2";
        "אחר הצהריים",
        "evening1";
        "ערב",
        "night1";
        "לילה",
        "night2";
        "לפנות בוקר";
    }
}
"stand-alone";
{
    "abbreviated";
    {
        "midnight";
        "חצות",
        "am";
        "לפנה"צ",
        "pm";
        "אחה"צ",
        "morning1";
        "בוקר",
        "afternoon1";
        "צהריים",
        "afternoon2";
        "אחר הצהריים",
        "evening1";
        "ערב",
        "night1";
        "לילה",
        "night2";
        "לפנות בוקר";
    }
    "narrow";
    {
        "midnight";
        "חצות",
        "am";
        "לפנה"צ",
        "pm";
        "אחה"צ",
        "morning1";
        "בוקר",
    }
}

```

```

        "afternoon1";
        "צהריים",
        "afternoon2";
        "אחר הצהריים",
        "evening1";
        "ערב",
        "night1";
        "לילה",
        "night2";
        "לפנות בוקר";
    }
    "wide";
    {
        "midnight";
        "חצות",
        "am";
        "לפנה"צ",
        "pm";
        "אחה"צ",
        "morning1";
        "בוקר",
        "afternoon1";
        "צהריים",
        "afternoon2";
        "אחר הצהריים",
        "evening1";
        "ערב",
        "night1";
        "לילה",
        "night2";
        "לפנות בוקר";
    }
}
}
"eras";
{
    "eraNames";
    {
        "0";
        "לפני הספירה",
        "0-alt-variant";
        "לפנה"ס",
        "1";
        "לספירה",
        "1-alt-variant";
        "CE";
    }
    "eraAbbr";
    {
        "0";
        "לפנה"ס",
        "0-alt-variant";
        "BCE",
        "1";
        "לספירה",
        "1-alt-variant";
        "CE";
    }
}

```



```

    }
    "eraNarrow";
    {
        "0";
        "לפנה"ס",
        "0-alt-variant";
        "BCE",
        "1";
        "לספירה",
        "1-alt-variant";
        "CE";
    }
}
"dateFormats";
{
    "full";
    "EEEE, d MMMM y",
    "long";
    "d MMMM y",
    "medium";
    "d MMM y",
    "short";
    "d.M.y";
}
"timeFormats";
{
    "full";
    "H:mm:ss zzzz",
    "long";
    "H:mm:ss z",
    "medium";
    "H:mm:ss",
    "short";
    "H:mm";
}
"dateTimeFormats";
{
    "full";
    "{1} בשעה {0}",
    "long";
    "{1} בשעה {0}",
    "medium";
    "{1}, {0}",
    "short";
    "{1}, {0}",
    "availableFormats";
    {
        "d";
        "d",
        "E";
        "ccc",
        "Ed";
        "E ה-d",
        "Ehm";
        "E h:mm a",
        "EHm";
        "E H:mm",
    }
}

```

```

        "Ehms";
        "E h:mm:ss a",
        "EHms";
        "E H:mm:ss",
        "Gy";
        "y G",
        "GyMMM";
        "MMM y G",
        "GyMMMd";
        "d 1MMM y G",
        "GyMMMED";
        "E, d 1MMM y G",
        "h";
        "h a",
        "H";
        "H",
        "hm";
        "h:mm a",
        "Hm";
        "H:mm",
        "hms";
        "h:mm:ss a",
        "Hms";
        "H:mm:ss",
        "hmsv";
        "h:mm:ss a v",
        "Hmsv";
        "HH:mm:ss v",
        "hmv";
        "h:mm a v",
        "Hmv";
        "HH:mm v",
        "M";
        "L",
        "Md";
        "d.M",
        "MED";
        "E, d.M",
        "MMM";
        "LLL",
        "MMMd";
        "d 1MMM",
        "MMMED";
        "E, d 1MMM",
        "MMMMd";
        "d 1MMMM",
        "MMMMW-count-one";
        "שבוע W 1MMM",
        "MMMMW-count-two";
        "שבוע W 2MMM",
        "MMMMW-count-many";
        "שבוע W 3MMM",
        "MMMMW-count-other";
        "שבוע W 4MMM",
        "ms";
        "mm:ss",
        "y";

```

```

        "y",
        "yM";
        "M.y",
        "yMd";
        "d.M.y",
        "yMEd";
        "E, d.M.y",
        "yMM";
        "M.y",
        "yMMM";
        "MMM y",
        "yMMMd";
        "d ıMMM y",
        "yMMMEd";
        "E, d ıMMM y",
        "yMMMM";
        "MMMM y",
        "yQQQ";
        "QQQ y",
        "yQQQQ";
        "QQQQ y",
        "yw-count-one";
        "שבוע y בשנת w שבוע",
        "yw-count-two";
        "שבוע y בשנת w שבוע",
        "yw-count-many";
        "שבוע y בשנת w שבוע",
        "yw-count-other";
        "שבוע y בשנת w שבוע";
    }
    "appendItems";
    {
        "Day";
        "{0} ({2}: {1})",
        "Day-Of-Week";
        "{0} {1}",
        "Era";
        "{1} {0}",
        "Hour";
        "{0} ({2}: {1})",
        "Minute";
        "{0} ({2}: {1})",
        "Month";
        "{0} ({2}: {1})",
        "Quarter";
        "{0} ({2}: {1})",
        "Second";
        "{0} ({2}: {1})",
        "Timezone";
        "{0} {1}",
        "Week";
        "{0} ({2}: {1})",
        "Year";
        "{1} {0}";
    }
    "intervalFormats";
    {

```

```
"intervalFormatFallback";
"{0} - {1}",
  "d";
{
  "d";
  "d-d";
}
"h";
{
  "a";
  "h a - h a",
    "h";
  "h-h a";
}
"H";
{
  "H";
  "H-H";
}
"hm";
{
  "a";
  "h:mm a - h:mm a",
    "h";
  "h:mm-h:mm a",
    "m";
  "h:mm-h:mm a";
}
"Hm";
{
  "H";
  "H:mm-H:mm",
    "m";
  "H:mm-H:mm";
}
"hm v";
{
  "a";
  "h:mm a - h:mm a v",
    "h";
  "h:mm-h:mm a v",
    "m";
  "h:mm-h:mm a v";
}
"Hm v";
{
  "H";
  "H:mm-H:mm v",
    "m";
  "H:mm-H:mm v";
}
"hv";
{
  "a";
  "h a - h a v",
    "h";
  "h-h a v";
```

```

    }
    "Hv";
    {
        "H";
        "H-H v";
    }
    "M";
    {
        "M";
        "M-M";
    }
    "Md";
    {
        "d";
        "d.M-d.M",
        "M";
        "d.M-d.M";
    }
    "MED";
    {
        "d";
        "EEEE d.M-EEEE d.M",
        "M";
        "EEEE d.M - EEEE d.M";
    }
    "MMM";
    {
        "M";
        "MMM-MMM";
    }
    "MMMd";
    {
        "d";
        "d-d  MMM",
        "M";
        "d  MMM-d  MMM";
    }
    "MMMED";
    {
        "d";
        "EEEE, d  MMM - EEEE, d  MMM",
        "M";
        "EEEE, d  MMM - EEEE, d  MMM";
    }
    "MMMM";
    {
        "M";
        "LLLL-LLLL";
    }
    "Y";
    {
        "Y";
        "Y-Y";
    }
    "YM";
    {
        "M";
    }

```

```

        "M.y-M.y",
        "y";
        "M.y-M.y";
    }
    "yMd";
    {
        "d";
        "dd.M.y - dd.M.y",
        "M";
        "d.M.y - d.M.y",
        "y";
        "d.M.y - d.M.y";
    }
    "yMEd";
    {
        "d";
        "EEEE d.M.y - EEEE d.M.y",
        "M";
        "EEEE d.M.y - EEEE d.M.y",
        "y";
        "EEEE d.M.y - EEEE d.M.y";
    }
    "yMMM";
    {
        "M";
        "MMM-MMM y",
        "y";
        "MMM y - MMM y";
    }
    "yMMMd";
    {
        "d";
        "d-d 1MMM y",
        "M";
        "d MMM - d MMM y",
        "y";
        "d MMM y - d MMM y";
    }
    "yMMMEd";
    {
        "d";
        "EEEE d MMM - EEEE d MMM y",
        "M";
        "EEEE d MMM - EEEE d MMM y",
        "y";
        "EEEE d MMM y - EEEE d MMM y";
    }
    "yMMMM";
    {
        "M";
        "MMMM-MMMM y",
        "y";
        "MMMM y-MMMM y";
    }
    }
}
}

```

```

    }
  }
}
}
}

```

INDEX.JSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
//Load the L10n, loadCldr from ej2-base
import { loadCldr, L10n } from '@syncfusion/ej2-base';
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
//load the CLDR data files.
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as heTimeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, heTimeZoneNames);
L10n.load({
  'he': {
    'daterangepicker': {
      applyText: 'להחיל טקסט',
      cancelText: 'בטל טקסט',
      customRange: 'טווח מותאם אישית',
      days: 'אִימִים',
      endLabel: 'ס',
      placeholder: 'בחר טווח',
      selectedDays: 'אִימִים נבחרים',
      startLabel: 'תוויית התחלה'
    }
  }
});
//import the datrangeepicker component
function App() {
  return <DateRangePickerComponent id="daterangepicker" locale='he'
enableRtl={true}/>;
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

import * as React from "react";
import * as ReactDOM from "react-dom";
//Load the L10n, loadCldr from ej2-base
import { loadCldr, L10n } from '@syncfusion/ej2-base';
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
//load the CLDR data files.
// Here we have referred local json files for preview purpose
import * as numberingSystems from './numberingSystems.json';
import * as gregorian from './ca-gregorian.json';
import * as numbers from './numbers.json';
import * as heTimeZoneNames from './timeZoneNames.json';
loadCldr(numberingSystems, gregorian, numbers, heTimeZoneNames);

```

```

L10n.load({
  'he': {
    'daterangepicker': {
      applyText: 'להחיל טקסט',
      cancelText: 'בטל טקסט',
      customRange: 'טווח מותאם אישית',
      days: 'ימים',
      endLabel: 'סוף',
      placeholder: 'בחר טווח',
      selectedDays: 'ימים נבחרים',
      startLabel: 'תחילת התחלה'
    }
  }
});
//import the datrangeepicker component
function App() {
  return <DateRangePickerComponent id="daterangepicker" locale='he'
enableRtl={true} />
};
ReactDOM.render(<App />, document.getElementById('element'));

```

NUMBERINGSYSTEMS.JSON

```

{
  "supplemental": {
    "version": {
      "_number": "$Revision: 12732 $",
      "_unicodeVersion": "9.0.0",
      "_cldrVersion": "31"
    },
    "numberingSystems": {
      "adlm": {
        "_digits": "ᲐᲑᲒᲓᲔᲕᲖᲗᲘᲙᲚᲛᲜᲝᲞᲟᲠᲡᲢᲣᲤᲥᲦᲧᲨᲩᲪᲫᲬᲭᲮᲯᲰᲱᲲᲳᲴᲵᲶᲷᲸᲹᲺ᲻᲼ᲽᲾᲿ",
        "_type": "numeric"
      },
      "ahom": {
        "_digits": "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍᱎᱏ᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱹᱺᱛᱟᱠᱣᱚᱰᱚᱨ",
        "_type": "numeric"
      },
      "arab": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "arabext": {
        "_digits": "٠١٢٣٤٥٦٧٨٩",
        "_type": "numeric"
      },
      "armn": {
        "_rules": "armenian-upper",
        "_type": "algorithmic"
      },
      "armnlow": {
        "_rules": "armenian-lower",
        "_type": "algorithmic"
      },
      "bali": {

```



```

    "_digits": "□□□□□□□□",
    "_type": "numeric"
  },
  "beng": {
    "_digits": "০১২৩৪৫৬৭৮৯",
    "_type": "numeric"
  },
  "bhks": {
    "_digits": "□□□□□□□□",
    "_type": "numeric"
  },
  "brah": {
    "_digits": "·ᱠᱡᱢᱫᱷᱚᱨᱵᱽᱫᱷ",
    "_type": "numeric"
  },
  "cakm": {
    "_digits": "᐀ᐁᐂᐃᐄᐅᐆᐇᐈ",
    "_type": "numeric"
  },
  "cham": {
    "_digits": "□□□□□□□□",
    "_type": "numeric"
  },
  "cyr1": {
    "_rules": "cyrillic-lower",
    "_type": "algorithmic"
  },
  "deva": {
    "_digits": "०१२३४५६७८९",
    "_type": "numeric"
  },
  "ethi": {
    "_rules": "ethiopic",
    "_type": "algorithmic"
  },
  "fullwide": {
    "_digits": "0 1 2 3 4 5 6 7 8 9",
    "_type": "numeric"
  },
  "geor": {
    "_rules": "georgian",
    "_type": "algorithmic"
  },
  "grek": {
    "_rules": "greek-upper",
    "_type": "algorithmic"
  },
  "greklow": {
    "_rules": "greek-lower",
    "_type": "algorithmic"
  },
  "gujr": {
    "_digits": "૦૧૨૩૪૫૬૭૮૯",
    "_type": "numeric"
  },

```

```

    "guru": {
      "_digits": "୦୧୨୩୪୫୬୭୮୯",
      "_type": "numeric"
    },
    "hanidays": {
      "_rules": "zh/SpelloutRules/spellout-numbering-days",
      "_type": "algorithmic"
    },
    "hanidec": {
      "_digits": "〇一二三四五六七八九",
      "_type": "numeric"
    },
    "hans": {
      "_rules": "zh/SpelloutRules/spellout-cardinal",
      "_type": "algorithmic"
    },
    "hansfin": {
      "_rules": "zh/SpelloutRules/spellout-cardinal-financial",
      "_type": "algorithmic"
    },
    "hant": {
      "_rules": "zh_Hant/SpelloutRules/spellout-cardinal",
      "_type": "algorithmic"
    },
    "hantfin": {
      "_rules": "zh_Hant/SpelloutRules/spellout-cardinal-financial",
      "_type": "algorithmic"
    },
    "hebr": {
      "_rules": "hebrew",
      "_type": "algorithmic"
    },
    "hmng": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    "java": {
      "_digits": "୦୧୨୩୪୫୬୭୮୯୦୧୨୩୪୫୬୭୮୯",
      "_type": "numeric"
    },
    "jpan": {
      "_rules": "ja/SpelloutRules/spellout-cardinal",
      "_type": "algorithmic"
    },
    "jpanfin": {
      "_rules": "ja/SpelloutRules/spellout-cardinal-financial",
      "_type": "algorithmic"
    },
    "kali": {
      "_digits": "□□□□□□□□□□",
      "_type": "numeric"
    },
    "khr": {
      "_digits": "០១២៣៤៥៦៧៨៩",
      "_type": "numeric"
    }
  }

```

```

},
"knda": {
  "_digits": "೦೧೨೩೪೫೬೭೮೯",
  "_type": "numeric"
},
"lana": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"lanatham": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"laoo": {
  "_digits": "໐໑໒໓໔໕໖໗໘໑",
  "_type": "numeric"
},
"latn": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"lepc": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"limb": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"mathbold": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathdbl": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathmono": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathsanb": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mathsans": {
  "_digits": "0123456789",
  "_type": "numeric"
},
"mlym": {
  "_digits": "൦൧൨൩൪൫൬൭൮൯",
  "_type": "numeric"
},
"modi": {
  "_digits": "□□□□□□□□□□",

```

```
    "_type": "numeric"
  },
  "mong": {
    "_digits": "᠓᠐᠐᠐᠐᠐᠐᠐᠐᠐",
    "_type": "numeric"
  },
  "mroo": {
    "_digits": "᠐᠐᠐᠐᠐᠐᠐᠐᠐᠐",
    "_type": "numeric"
  },
  "mtei": {
    "_digits": "᠐᠑᠑᠑᠑᠑᠑᠑᠑᠑᠑",
    "_type": "numeric"
  },
  "mymr": {
    "_digits": "᠐᠑᠑᠑᠑᠑᠑᠑᠑᠑",
    "_type": "numeric"
  },
  "mymrshan": {
    "_digits": "᠐1᠒᠕᠕᠔᠑᠘88",
    "_type": "numeric"
  },
  "mymrtlng": {
    "_digits": "᠐᠑᠑᠑᠑᠑᠑᠑᠑᠑᠑",
    "_type": "numeric"
  },
  "newa": {
    "_digits": "᠐᠐᠐᠐᠐᠐᠐᠐᠐᠐",
    "_type": "numeric"
  },
  "nkoo": {
    "_digits": "᠑᠑᠑᠑᠑᠑᠑᠑᠑᠑",
    "_type": "numeric"
  },
  "olck": {
    "_digits": "᠐᠑᠑᠑᠑᠑᠑᠑᠑᠑",
    "_type": "numeric"
  },
  "orya": {
    "_digits": "᠐᠑᠑᠑᠑᠑᠑᠑᠑᠑",
    "_type": "numeric"
  },
  "osma": {
    "_digits": "᠐᠑᠑᠑᠑᠑᠑᠑᠑᠑",
    "_type": "numeric"
  },
  "roman": {
    "_rules": "roman-upper",
    "_type": "algorithmic"
  },
  "romanlow": {
    "_rules": "roman-lower",
    "_type": "algorithmic"
  }
```

```

},
"saur": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"shrd": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"sind": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"sinh": {
  "_digits": "⁂௧௧௧௧௧௧௧௧௧௧௧௧",
  "_type": "numeric"
},
"sora": {
  "_digits": "ᱚᱠᱟᱨᱚᱸᱰᱟᱦᱟᱫ",
  "_type": "numeric"
},
"sund": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"takr": {
  "_digits": "□□□□□□□□□□",
  "_type": "numeric"
},
"talv": {
  "_digits": "ᲠᲚᲛᲗᲡᲣᲤᲧᲨᲩᲪᲫᲬᲭᲮᲯᲰᲱᲲᲳᲴᲵᲶᲷᲸᲹᲺ᲻᲼ᲽᲾᲿ",
  "_type": "numeric"
},
"tml": {
  "_rules": "tamil",
  "_type": "algorithmic"
},
"tmldec": {
  "_digits": "௦௧௨௩௪௫௬௭௮௯",
  "_type": "numeric"
},
"telu": {
  "_digits": "౦౧౨౩౪౫౬౭౮౯",
  "_type": "numeric"
},
"thai": {
  "_digits": "๐๑๒๓๔๕๖๗๘๙",
  "_type": "numeric"
},
"tib": {
  "_digits": "༠༡༢༣༤༥༦༧༨༩",
  "_type": "numeric"
},
"tirh": {
  "_digits": "□□□□□□□□□□",

```

```
{
    "_type": "numeric"
},
{
    "vaii": {
        "_digits": "ᱵᱟᱹᱨᱩᱝᱜᱚᱴᱷᱟᱲᱤ",
        "_type": "numeric"
    },
    "wara": {
        "_digits": "ᱠᱟᱢᱦᱮᱸᱰᱤᱡᱽᱫᱷᱟᱲᱤ",
        "_type": "numeric"
    }
}
}
```

NUMBERINGSYSTEMS.JSX

```
{
  "supplemental";
  {
    "version";
    {
      "_number";
      "$Revision: 12732 $",
      "_unicodeVersion";
      "9.0.0",
      "_cldrVersion";
      "31";
    }
    "numberingSystems";
    {
      "adlm";
      {
        "_digits";
        "ᲀᲁᲂᲃᲄᲅᲆᲇᲈᲉᲐᲑᲒᲓᲔᲕᲖᲗᲘᲙᲚᲛᲜᲝᲞᲟᲠᲡᲢᲣᲤᲥᲦᲧᲨᲩᲪᲫᲬᲭᲮᲯᲰᲱᲲᲳᲴᲵᲶᲷᲸᲹᲺ᲻᲼ᲽᲾᲿ",
        "_type";
        "numeric";
      }
      "ahom";
      {
        "_digits";
        "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱰᱱᱲᱳᱴᱵᱶᱷᱸᱹᱺᱻᱼᱽ᱾᱿",
        "_type";
        "numeric";
      }
      "arab";
      {
        "_digits";
        "٠١٢٣٤٥٦٧٨٩",
        "_type";
        "numeric";
      }
      "arabext";
      {
        "_digits";
        "٠١٢٣٤٥٦٧٨٩",
        "_type";

```

```

        "numeric";
    }
    "armn";
    {
        "_rules";
        "armenian-upper",
        "_type";
        "algorithmic";
    }
    "armnlow";
    {
        "_rules";
        "armenian-lower",
        "_type";
        "algorithmic";
    }
    "bali";
    {
        "_digits";
        "ᮀᮁᮂᮃᮄᮅᮆᮇᮈᮉ",
        "_type";
        "numeric";
    }
    "beng";
    {
        "_digits";
        "০১২৩৪৫৬৭৮৯",
        "_type";
        "numeric";
    }
    "bhks";
    {
        "_digits";
        "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉",
        "_type";
        "numeric";
    }
    "brah";
    {
        "_digits";
        "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉",
        "_type";
        "numeric";
    }
    "cakm";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";
        "numeric";
    }
    "cham";
    {
        "_digits";
        "᐀ᐁᐂᐃᐄᐅᐆᐇᐈᐉ",
        "_type";

```

```
        "numeric";
    }
    "cyrl";
    {
        "_rules";
        "cyrillic-lower",
        "_type";
        "algorithmic";
    }
    "deva";
    {
        "_digits";
        "०१२३४५६७८९",
        "_type";
        "numeric";
    }
    "ethi";
    {
        "_rules";
        "ethiopic",
        "_type";
        "algorithmic";
    }
    "fullwide";
    {
        "_digits";
        "0 1 2 3 4 5 6 7 8 9",
        "_type";
        "numeric";
    }
    "geor";
    {
        "_rules";
        "georgian",
        "_type";
        "algorithmic";
    }
    "grek";
    {
        "_rules";
        "greek-upper",
        "_type";
        "algorithmic";
    }
    "greklow";
    {
        "_rules";
        "greek-lower",
        "_type";
        "algorithmic";
    }
    "gujr";
    {
        "_digits";
        "૦૧૨૩૪૫૬૭૮૯",
        "_type";
```



```

        "numeric";
    }
    "guru";
    {
        "_digits";
        "୦୧୨୩୪୫୬୭୮୯",
        "_type";
        "numeric";
    }
    "hanidays";
    {
        "_rules";
        "zh/SpelloutRules/spellout-numbering-days",
        "_type";
        "algorithmic";
    }
    "hanidec";
    {
        "_digits";
        "〇一二三四五六七八九",
        "_type";
        "numeric";
    }
    "hans";
    {
        "_rules";
        "zh/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "hansfin";
    {
        "_rules";
        "zh/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "hant";
    {
        "_rules";
        "zh_Hant/SpelloutRules/spellout-cardinal",
        "_type";
        "algorithmic";
    }
    "hantfin";
    {
        "_rules";
        "zh_Hant/SpelloutRules/spellout-cardinal-financial",
        "_type";
        "algorithmic";
    }
    "hebr";
    {
        "_rules";
        "hebrew",
        "_type";
    }

```

```

"algorithmic";
}
"hmng";
{
  "_digits";
  "၀၀၀၀၀၀၀၀၀၀",
  "_type";
  "numeric";
}
"java";
{
  "_digits";
  "၀၀၀၀၀၀၀၀၀၀၀၀၀၀၀၀၀၀၀၀၀၀",
  "_type";
  "numeric";
}
"jpan";
{
  "_rules";
  "ja/SpelloutRules/spellout-cardinal",
  "_type";
  "algorithmic";
}
"jpanfin";
{
  "_rules";
  "ja/SpelloutRules/spellout-cardinal-financial",
  "_type";
  "algorithmic";
}
"kali";
{
  "_digits";
  "၀၀၀၀၀၀၀၀၀၀၀၀",
  "_type";
  "numeric";
}
"khmr";
{
  "_digits";
  "០១២៣៤៥៦៧៨៩",
  "_type";
  "numeric";
}
"knda";
{
  "_digits";
  "೦೧೨೩೪೫೬೭೮೯",
  "_type";
  "numeric";
}
"lana";
{
  "_digits";
  "၀၀၀၀၀၀၀၀၀၀",

```

```

        "_type";
        "numeric";
    }
    "lanatham";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "laoo";
    {
        "_digits";
        "໐໑໒໓໔໕໖໗໘໑",
        "_type";
        "numeric";
    }
    "latn";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "lepc";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "limb";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "mathbold";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathdbl";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathmono";
    {
        "_digits";
        "0123456789",

```

```

        "_type";
        "numeric";
    }
    "mathsanb";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mathsans";
    {
        "_digits";
        "0123456789",
        "_type";
        "numeric";
    }
    "mlym";
    {
        "_digits";
        "ഘറന൪റ൬൮൯",
        "_type";
        "numeric";
    }
    "modi";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "mong";
    {
        "_digits";
        "᠎ᠠᠨᠣᠭ",
        "_type";
        "numeric";
    }
    "mroo";
    {
        "_digits";
        "□□□□□□□□",
        "_type";
        "numeric";
    }
    "mtei";
    {
        "_digits";
        "ၿၿၿၿၿၿၿၿ",
        "_type";
        "numeric";
    }
    "mymr";
    {
        "_digits";

```

```

        "၀၁၂၃၄၅၆၇၈",
        "_type";
        "numeric";
    }
    "mymrshan";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈",
        "_type";
        "numeric";
    }
    "mymrtlng";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈",
        "_type";
        "numeric";
    }
    "newa";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈",
        "_type";
        "numeric";
    }
    "nkoo";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈",
        "_type";
        "numeric";
    }
    "olck";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈",
        "_type";
        "numeric";
    }
    "orya";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈",
        "_type";
        "numeric";
    }
    "osma";
    {
        "_digits";
        "၀၁၂၃၄၅၆၇၈",
        "_type";
        "numeric";
    }
    "roman";

```

```

    {
      "_rules";
      "roman-upper",
      "_type";
      "algorithmic";
    }
    "romanlow";
    {
      "_rules";
      "roman-lower",
      "_type";
      "algorithmic";
    }
    "saur";
    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "shrd";
    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "sind";
    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "sinh";
    {
      "_digits";
      "𑆑𑆒𑆓𑆔𑆕𑆖𑆗𑆘𑆙𑆚",
      "_type";
      "numeric";
    }
    "sora";
    {
      "_digits";
      "᱀᱁᱂᱃᱄᱅᱆᱇᱈᱉᱊᱋᱌ᱍᱎᱏ᱐᱑᱒᱓᱔᱕᱖᱗᱘᱙ᱚᱛᱜᱝᱞᱟᱠᱡᱢᱣᱤᱥᱦᱧᱨᱩᱪᱫᱬᱭᱮᱹᱺᱻᱼᱽ᱾᱿",
      "_type";
      "numeric";
    }
    "sund";
    {
      "_digits";
      "□□□□□□□□",
      "_type";
      "numeric";
    }
    "takr";

```

```

    {
      "_digits";
      "□□□□□□□□□□",
      "_type";
      "numeric";
    }
    "tal";
    {
      "_digits";
      "ᱠᱡᱷᱚᱸᱰ",
      "_type";
      "numeric";
    }
    "tam";
    {
      "_rules";
      "tamil",
      "_type";
      "algorithmic";
    }
    "tamldc";
    {
      "_digits";
      "௦௧௨௩௪௫௬௭௮௯",
      "_type";
      "numeric";
    }
    "tel";
    {
      "_digits";
      "౦౧౨౩౪౫౬౭౮౯",
      "_type";
      "numeric";
    }
    "thai";
    {
      "_digits";
      "๐๑๒๓๔๕๖๗๘๙",
      "_type";
      "numeric";
    }
    "tib";
    {
      "_digits";
      "༠༡༢༣༤༥༦༧༨༩",
      "_type";
      "numeric";
    }
    "tirh";
    {
      "_digits";
      "□□□□□□□□□□",
      "_type";
      "numeric";
    }
    "vai";

```

```
{
    "_digits";
    "ୱ|୧୨୩୪୫୬୭୮୯",
    "_type";
    "numeric";
}
"wara";
{
    "_digits";
    "□□□□□□□□",
    "_type";
    "numeric";
}
}
```

NUMBERS.JSON

```
{
  "main": {
    "he": {
      "identity": {
        "version": {
          "_number": "$Revision: 13259 $",
          "_cldrVersion": "31.0.1"
        },
        "language": "he"
      },
      "numbers": {
        "defaultNumberingSystem": "latn",
        "otherNumberingSystems": {
          "native": "latn",
          "traditional": "hebr"
        },
        "minimumGroupingDigits": "1",
        "symbols-numberSystem-latn": {
          "decimal": ".",
          "group": ",",
          "list": ";",
          "percentSign": "%",
          "plusSign": "+",
          "minusSign": "-",
          "exponential": "E",
          "superscriptingExponent": "×",
          "perMille": "‰",
          "infinity": "∞",
          "nan": "NaN",
          "timeSeparator": ":"
        },
        "decimalFormats-numberSystem-latn": {
          "standard": "#,##0.###",
          "long": {
            "decimalFormat": {
              "1000-count-one": "אלף 0",
              "1000-count-two": "אלף 0",

```



```

    "1000-count-many": "אלף 0",
    "1000-count-other": "אלף 0",
    "10000-count-one": "אלף 00",
    "10000-count-two": "אלף 00",
    "10000-count-many": "אלף 00",
    "10000-count-other": "אלף 00",
    "100000-count-one": "אלף 000",
    "100000-count-two": "אלף 000",
    "100000-count-many": "אלף 000",
    "100000-count-other": "אלף 000",
    "1000000-count-one": "מיליון 0",
    "1000000-count-two": "מיליון 0",
    "1000000-count-many": "מיליון 0",
    "1000000-count-other": "מיליון 0",
    "10000000-count-one": "מיליון 00",
    "10000000-count-two": "מיליון 00",
    "10000000-count-many": "מיליון 00",
    "10000000-count-other": "מיליון 00",
    "100000000-count-one": "מיליון 000",
    "100000000-count-two": "מיליון 000",
    "100000000-count-many": "מיליון 000",
    "100000000-count-other": "מיליון 000",
    "1000000000-count-one": "מיליארד 0",
    "1000000000-count-two": "מיליארד 0",
    "1000000000-count-many": "מיליארד 0",
    "1000000000-count-other": "מיליארד 0",
    "10000000000-count-one": "מיליארד 00",
    "10000000000-count-two": "מיליארד 00",
    "10000000000-count-many": "מיליארד 00",
    "10000000000-count-other": "מיליארד 00",
    "100000000000-count-one": "מיליארד 000",
    "100000000000-count-two": "מיליארד 000",
    "100000000000-count-many": "מיליארד 000",
    "100000000000-count-other": "מיליארד 000",
    "1000000000000-count-one": "טריליון 0",
    "1000000000000-count-two": "טריליון 0",
    "1000000000000-count-many": "טריליון 0",
    "1000000000000-count-other": "טריליון 0",
    "10000000000000-count-one": "טריליון 00",
    "10000000000000-count-two": "טריליון 00",
    "10000000000000-count-many": "טריליון 00",
    "10000000000000-count-other": "טריליון 00",
    "100000000000000-count-one": "טריליון 000",
    "100000000000000-count-two": "טריליון 000",
    "100000000000000-count-many": "טריליון 000",
    "100000000000000-count-other": "טריליון 000"
  },
  "short": {
    "decimalFormat": {
      "1000-count-one": "0K",
      "1000-count-two": "0K",
      "1000-count-many": "0K",
      "1000-count-other": "0K",
      "10000-count-one": "00K",
      "10000-count-two": "00K",
      "10000-count-many": "00K",
    }
  }
}

```

```

        "10000-count-other": "00K",
        "100000-count-one": "000K",
        "100000-count-two": "000K",
        "100000-count-many": "000K",
        "100000-count-other": "000K",
        "1000000-count-one": "0M",
        "1000000-count-two": "0M",
        "1000000-count-many": "0M",
        "1000000-count-other": "0M",
        "10000000-count-one": "00M",
        "10000000-count-two": "00M",
        "10000000-count-many": "00M",
        "10000000-count-other": "00M",
        "100000000-count-one": "000M",
        "100000000-count-two": "000M",
        "100000000-count-many": "000M",
        "100000000-count-other": "000M",
        "1000000000-count-one": "0B",
        "1000000000-count-two": "0B",
        "1000000000-count-many": "0B",
        "1000000000-count-other": "0B",
        "10000000000-count-one": "00B",
        "10000000000-count-two": "00B",
        "10000000000-count-many": "00B",
        "10000000000-count-other": "00B",
        "100000000000-count-one": "0T",
        "100000000000-count-two": "0T",
        "100000000000-count-many": "0T",
        "100000000000-count-other": "0T",
        "1000000000000-count-one": "00T",
        "1000000000000-count-two": "00T",
        "1000000000000-count-many": "00T",
        "1000000000000-count-other": "00T",
        "10000000000000-count-one": "000T",
        "10000000000000-count-two": "000T",
        "10000000000000-count-many": "000T",
        "10000000000000-count-other": "000T"
    }
}
},
"scientificFormats-numberSystem-latn": {
    "standard": "#E0"
},
"percentFormats-numberSystem-latn": {
    "standard": "#,##0%"
},
"currencyFormats-numberSystem-latn": {
    "currencySpacing": {
        "beforeCurrency": {
            "currencyMatch": "[:^S:]",
            "surroundingMatch": "[:digit:]",
            "insertBetween": " "
        }
    }
},

```

```

    "afterCurrency": {
      "currencyMatch": "[:^S:]",
      "surroundingMatch": "[:digit:]",
      "insertBetween": " "
    }
  },
  "standard": "##0.00#, -¤; ##0.00#, ¤",
  "accounting": "#, ##0.00 ¤",
  "short": {
    "standard": {
      "1000-count-one": "¤ 0K",
      "1000-count-two": "¤ 0K",
      "1000-count-many": "¤0K",
      "1000-count-other": "¤ 0K",
      "10000-count-one": "¤00K",
      "10000-count-two": "¤00K",
      "10000-count-many": "¤00K",
      "10000-count-other": "¤ 00K",
      "100000-count-one": "¤000K",
      "100000-count-two": "¤000K",
      "100000-count-many": "¤000K",
      "100000-count-other": "¤000K",
      "1000000-count-one": "¤0M",
      "1000000-count-two": "¤0M",
      "1000000-count-many": "¤0M",
      "1000000-count-other": "¤0M",
      "10000000-count-one": "¤00M",
      "10000000-count-two": "¤00M",
      "10000000-count-many": "¤00M",
      "10000000-count-other": "¤00M",
      "100000000-count-one": "¤000M",
      "100000000-count-two": "¤000M",
      "100000000-count-many": "¤000M",
      "100000000-count-other": "¤000M",
      "1000000000-count-one": "¤0B",
      "1000000000-count-two": "¤0B",
      "1000000000-count-many": "¤0B",
      "1000000000-count-other": "¤0B",
      "10000000000-count-one": "¤00B",
      "10000000000-count-two": "¤00B",
      "10000000000-count-many": "¤00B",
      "10000000000-count-other": "¤00B",
      "100000000000-count-one": "¤000B",
      "100000000000-count-two": "¤000B",
      "100000000000-count-many": "¤000B",
      "100000000000-count-other": "¤000B",
      "1000000000000-count-one": "¤0T",
      "1000000000000-count-two": "¤0T",
      "1000000000000-count-many": "¤0T",
      "1000000000000-count-other": "¤0T",
      "10000000000000-count-one": "¤00T",
      "10000000000000-count-two": "¤00T",
      "10000000000000-count-many": "¤00T",
      "10000000000000-count-other": "¤00T",
      "100000000000000-count-one": "¤000T",
      "100000000000000-count-two": "¤000T",
      "100000000000000-count-many": "¤000T",
    }
  }
}

```

```

        "100000000000000-count-other": "א000T"
    },
    },
    "unitPattern-count-one": "{0} {1}",
    "unitPattern-count-two": "{0} {1}",
    "unitPattern-count-many": "{0} {1}",
    "unitPattern-count-other": "{0} {1}"
},
"miscPatterns-numberSystem-latn": {
    "atLeast": "≥{0}+",
    "range": "{0}-{1}"
},
"minimalPairs": {
    "pluralMinimalPairs": "שנה",
    "pluralMinimalPairs": "שנתיים",
    "pluralMinimalPairs": "{0} שנה",
    "pluralMinimalPairs": "{0} שנים",
    "other": "פנה ימינה בפנייה ה-{0}"
}
}
}
}
}
}

```

NUMBERS.JSX

```

{
    "main";
    {
        "he";
        {
            "identity";
            {
                "version";
                {
                    "_number";
                    "$Revision: 13259 $",
                    "_cldrVersion";
                    "31.0.1";
                }
                "language";
                "he";
            }
            "numbers";
            {
                "defaultNumberingSystem";
                "latn",
                "otherNumberingSystems";
                {
                    "native";
                    "latn",
                    "traditional";
                    "hebr";
                }
                "minimumGroupingDigits";
                "1",
            }
        }
    }
}

```

```

        "symbols-numberSystem-latn";
    {
        "decimal";
        ".",
        "group";
        ",",
        "list";
        ";",
        "percentSign";
        "%",
        "plusSign";
        "+",
        "minusSign";
        "-",
        "exponential";
        "E",
        "superscriptingExponent";
        "×",
        "perMille";
        "‰",
        "infinity";
        "∞",
        "nan";
        "NaN",
        "timeSeparator";
        ":";
    }
    "decimalFormats-numberSystem-latn";
    {
        "standard";
        "#,##0.###",
        "long";
        {
            "decimalFormat";
            {
                "1000-count-one";
                "q̌ʌ 0",
                "1000-count-two";
                "q̌ʌ 0",
                "1000-count-many";
                "q̌ʌ 0",
                "1000-count-other";
                "q̌ʌ 0",
                "10000-count-one";
                "q̌ʌ 00",
                "10000-count-two";
                "q̌ʌ 00",
                "10000-count-many";
                "q̌ʌ 00",
                "10000-count-other";
                "q̌ʌ 00",
                "100000-count-one";
                "q̌ʌ 000",
                "100000-count-two";
                "q̌ʌ 000",
                "100000-count-many";
                "q̌ʌ 000",
            }
        }
    }

```

```
"100000-count-other";
"אלף 000",
"1000000-count-one";
"0 מיליון",
"1000000-count-two";
"0 מיליון",
"1000000-count-many";
"0 מיליון",
"1000000-count-other";
"0 מיליון",
"10000000-count-one";
"00 מיליון",
"10000000-count-two";
"00 מיליון",
"10000000-count-many";
"00 מיליון",
"10000000-count-other";
"00 מיליון",
"100000000-count-one";
"000 מיליון",
"100000000-count-two";
"000 מיליון",
"100000000-count-many";
"000 מיליון",
"100000000-count-other";
"000 מיליון",
"1000000000-count-one";
"0 מיליארד",
"1000000000-count-two";
"0 מיליארד",
"1000000000-count-many";
"0 מיליארד",
"1000000000-count-other";
"0 מיליארד",
"10000000000-count-one";
"00 מיליארד",
"10000000000-count-two";
"00 מיליארד",
"10000000000-count-many";
"00 מיליארד",
"10000000000-count-other";
"00 מיליארד",
"100000000000-count-one";
"000 מיליארד",
"100000000000-count-two";
"000 מיליארד",
"100000000000-count-many";
"000 מיליארד",
"100000000000-count-other";
"000 מיליארד",
"1000000000000-count-one";
"0 טריליון",
"1000000000000-count-two";
"0 טריליון",
"1000000000000-count-many";
"0 טריליון",
"1000000000000-count-other";
```

```

        "טריליון 0",
        "10000000000000-count-one";
        "טריליון 00",
        "10000000000000-count-two";
        "טריליון 00",
        "10000000000000-count-many";
        "טריליון 00",
        "10000000000000-count-other";
        "טריליון 00",
        "10000000000000-count-one";
        "טריליון 000",
        "100000000000000-count-two";
        "טריליון 000",
        "100000000000000-count-many";
        "טריליון 000",
        "100000000000000-count-other";
        "טריליון 000";
    }
}
"short";
{
    "decimalFormat";
    {
        "1000-count-one";
        "0K",
        "1000-count-two";
        "0K",
        "1000-count-many";
        "0K",
        "1000-count-other";
        "0K",
        "10000-count-one";
        "00K",
        "10000-count-two";
        "00K",
        "10000-count-many";
        "00K",
        "10000-count-other";
        "00K",
        "100000-count-one";
        "000K",
        "100000-count-two";
        "000K",
        "100000-count-many";
        "000K",
        "100000-count-other";
        "000K",
        "1000000-count-one";
        "0M",
        "1000000-count-two";
        "0M",
        "1000000-count-many";
        "0M",
        "1000000-count-other";
        "0M",
        "10000000-count-one";
        "00M",
    }
}

```

```
"10000000-count-two";
"00M",
"10000000-count-many";
"00M",
"10000000-count-other";
"00M",
"10000000-count-one";
"000M",
"10000000-count-two";
"000M",
"10000000-count-many";
"000M",
"10000000-count-other";
"000M",
"10000000-count-one";
"0B",
"100000000-count-two";
"0B",
"100000000-count-many";
"0B",
"100000000-count-other";
"0B",
"1000000000-count-one";
"00B",
"1000000000-count-two";
"00B",
"1000000000-count-many";
"00B",
"1000000000-count-other";
"00B",
"10000000000-count-one";
"000B",
"10000000000-count-two";
"000B",
"10000000000-count-many";
"000B",
"10000000000-count-other";
"000B",
"100000000000-count-one";
"0T",
"100000000000-count-two";
"0T",
"100000000000-count-many";
"0T",
"100000000000-count-other";
"0T",
"1000000000000-count-one";
"00T",
"1000000000000-count-two";
"00T",
"1000000000000-count-many";
"00T",
"1000000000000-count-other";
"00T",
"10000000000000-count-one";
"000T",
"10000000000000-count-two";
```



```

        "000T",
        "1000000000000000-count-many";
        "000T",
        "1000000000000000-count-other";
        "000T";
    }
}
}
"scientificFormats-numberSystem-latn";
{
    "standard";
    "#E0";
}
"percentFormats-numberSystem-latn";
{
    "standard";
    "#,##0%";
}
"currencyFormats-numberSystem-latn";
{
    "currencySpacing";
    {
        "beforeCurrency";
        {
            "currencyMatch";
            "[:^S:]",
            "surroundingMatch";
            "[:digit:]",
            "insertBetween";
            " ";
        }
        "afterCurrency";
        {
            "currencyMatch";
            "[:^S:]",
            "surroundingMatch";
            "[:digit:]",
            "insertBetween";
            " ";
        }
    }
}
"standard";
"##0.00#,¤; ##0.00#, ¤",
    "accounting";
"#,##0.00 ¤",
    "short";
{
    "standard";
    {
        "1000-count-one";
        "¤ 0K",
        "1000-count-two";
        "¤ 0K",
        "1000-count-many";
        "¤0K",
        "1000-count-other";
        "¤ 0K",
    }
}

```

```
        "10000-count-one";
    "¤00K",
        "10000-count-two";
    "¤00K",
        "10000-count-many";
    "¤00K",
        "10000-count-other";
    "¤ 00K",
        "100000-count-one";
    "¤000K",
        "100000-count-two";
    "¤000K",
        "100000-count-many";
    "¤000K",
        "100000-count-other";
    "¤000K",
        "1000000-count-one";
    "¤0M",
        "1000000-count-two";
    "¤0M",
        "1000000-count-many";
    "¤0M",
        "1000000-count-other";
    "¤0M",
        "10000000-count-one";
    "¤00M",
        "10000000-count-two";
    "¤00M",
        "10000000-count-many";
    "¤00M",
        "10000000-count-other";
    "¤00M",
        "100000000-count-one";
    "¤000M",
        "100000000-count-two";
    "¤000M",
        "100000000-count-many";
    "¤000M",
        "100000000-count-other";
    "¤000M",
        "1000000000-count-one";
    "¤0B",
        "1000000000-count-two";
    "¤0B",
        "1000000000-count-many";
    "¤0B",
        "1000000000-count-other";
    "¤0B",
        "10000000000-count-one";
    "¤00B",
        "10000000000-count-two";
    "¤00B",
        "10000000000-count-many";
    "¤00B",
        "10000000000-count-other";
    "¤00B",
        "100000000000-count-one";
```

```

        "א000B",
        "100000000000-count-two";
        "א000B",
        "100000000000-count-many";
        "א000B",
        "100000000000-count-other";
        "א000B",
        "100000000000-count-one";
        "א0T",
        "100000000000-count-two";
        "א0T",
        "100000000000-count-many";
        "א0T",
        "100000000000-count-other";
        "א0T",
        "100000000000-count-one";
        "א00T",
        "100000000000-count-two";
        "א00T",
        "100000000000-count-many";
        "א00T",
        "100000000000-count-other";
        "א00T",
        "100000000000-count-one";
        "א000T",
        "100000000000-count-two";
        "א000T",
        "100000000000-count-many";
        "א000T",
        "100000000000-count-other";
        "א000T";
    }
}
"unitPattern-count-one";
"{0} {1}",
    "unitPattern-count-two";
"{0} {1}",
    "unitPattern-count-many";
"{0} {1}",
    "unitPattern-count-other";
"{0} {1}";
}
"miscPatterns-numberSystem-latn";
{
    "atLeast";
    "≥{0}+",
        "range";
    "{0}-{1}";
}
"minimalPairs";
{
    "pluralMinimalPairs";
    "שנה",
        "pluralMinimalPairs";
    "שנתיים",
        "pluralMinimalPairs";
    "{0} שנה",

```

TIMEZONENAMES.JSON

Copyright © 2001 -2024 Syncfusion Inc.

```

        "exemplarCity": "אושואיה"
    },
    "La_Rioja": {
        "exemplarCity": "לה ריוחה"
    },
    "San_Luis": {
        "exemplarCity": "סן לואיס"
    },
    "Salta": {
        "exemplarCity": "סלטה"
    },
    "Tucuman": {
        "exemplarCity": "טוקומן"
    }
},
"Aruba": {
    "exemplarCity": "ארובה"
},
"Asuncion": {
    "exemplarCity": "אסונסיון"
},
"Bahia": {
    "exemplarCity": "באהיה"
},
"Bahia_Banderas": {
    "exemplarCity": "באהיה בנדרס"
},
"Barbados": {
    "exemplarCity": "ברבדוס"
},
"Belem": {
    "exemplarCity": "בלם"
},
"Belize": {
    "exemplarCity": "בליז"
},
"Blanc-Sablon": {
    "exemplarCity": "בלאן-סבלון"
},
"Boa_Vista": {
    "exemplarCity": "בואה ויסטה"
},
"Bogota": {
    "exemplarCity": "בוגוטה"
},
"Boise": {
    "exemplarCity": "בויסי"
},
"Buenos_Aires": {
    "exemplarCity": "בואנוס איירס"
},
"Cambridge_Bay": {
    "exemplarCity": "קיימברידג' ביי"
},
"Campo_Grande": {
    "exemplarCity": "קמפו גרנדה"
},

```

```
"Cancun": {
  "exemplarCity": "קנקון"
},
"Caracas": {
  "exemplarCity": "קראקס"
},
"Catamarca": {
  "exemplarCity": "קטמרקה"
},
"Cayenne": {
  "exemplarCity": "קאייין"
},
"Cayman": {
  "exemplarCity": "קיימן"
},
"Chicago": {
  "exemplarCity": "שיקגו"
},
"Chihuahua": {
  "exemplarCity": "צ'יוואוואה"
},
"Coral_Harbour": {
  "exemplarCity": "אטיקוקן"
},
"Cordoba": {
  "exemplarCity": "קורדובה"
},
"Costa_Rica": {
  "exemplarCity": "קוסטה ריקה"
},
"Creston": {
  "exemplarCity": "קרטסון"
},
"Cuiaba": {
  "exemplarCity": "קויאבה"
},
"Curacao": {
  "exemplarCity": "קוראסאו"
},
"Danmarkshavn": {
  "exemplarCity": "דנמרקסהוון"
},
"Dawson": {
  "exemplarCity": "דוסון"
},
"Dawson_Creek": {
  "exemplarCity": "דוסון קריק"
},
"Denver": {
  "exemplarCity": "דנוור"
},
"Detroit": {
  "exemplarCity": "דטרויט"
},
"Dominica": {
  "exemplarCity": "דומיניקה"
},
},
```

```
"Edmonton": {
  "exemplarCity": "אדמונטון"
},
"Eirunepe": {
  "exemplarCity": "אירונפי"
},
"El_Salvador": {
  "exemplarCity": "אל סלבדור"
},
"Fort_Nelson": {
  "exemplarCity": "פורט נלסון"
},
"Fortaleza": {
  "exemplarCity": "פורטאלזה"
},
"Glace_Bay": {
  "exemplarCity": "גלייס ביי"
},
"Godthab": {
  "exemplarCity": "נואוק"
},
"Goose_Bay": {
  "exemplarCity": "גוס ביי"
},
"Grand_Turk": {
  "exemplarCity": "גרנד טורק"
},
"Grenada": {
  "exemplarCity": "גרנדה"
},
"Guadeloupe": {
  "exemplarCity": "גואדלופ"
},
"Guatemala": {
  "exemplarCity": "גואטמלה"
},
"Guayaquil": {
  "exemplarCity": "גואיאקיל"
},
"Guyana": {
  "exemplarCity": "גיאנה"
},
"Halifax": {
  "exemplarCity": "הליפקס"
},
"Havana": {
  "exemplarCity": "הוואנה"
},
"Hermosillo": {
  "exemplarCity": "הרמוסיו"
},
"Indiana": {
  "Vincennes": {
    "exemplarCity": "וינסנס, אינדיאנה"
  },
  "Petersburg": {
    "exemplarCity": "פיטרסבורג, אינדיאנה"
  }
}
```

```

    },
    "Tell_City": {
      "exemplarCity": "טל סיטי, אינדיאנה"
    },
    "Knox": {
      "exemplarCity": "נוקס, אינדיאנה"
    },
    "Winamac": {
      "exemplarCity": "ווינמאק, אינדיאנה"
    },
    "Marengo": {
      "exemplarCity": "מרנגו, אינדיאנה"
    },
    "Vevay": {
      "exemplarCity": "ויוואיי, אינדיאנה"
    }
  },
  "Indianapolis": {
    "exemplarCity": "אינדיאנפוליס"
  },
  "Inuvik": {
    "exemplarCity": "אינוויק"
  },
  "Iqaluit": {
    "exemplarCity": "איקלואיט"
  },
  "Jamaica": {
    "exemplarCity": "ג'מייקה"
  },
  "Jujuy": {
    "exemplarCity": "חוחוי"
  },
  "Juneau": {
    "exemplarCity": "ג'וננו"
  },
  "Kentucky": {
    "Monticello": {
      "exemplarCity": "מונטיצ'לו, קנטאקי"
    }
  },
  "Kralendijk": {
    "exemplarCity": "קרלנדייק"
  },
  "La_Paz": {
    "exemplarCity": "לה פאס"
  },
  "Lima": {
    "exemplarCity": "לימה"
  },
  "Los_Angeles": {
    "exemplarCity": "לוס אנג'לס"
  },
  "Louisville": {
    "exemplarCity": "לואיוויל"
  },
  "Lower_Princes": {
    "exemplarCity": "לואוור פרינסס קוורטר"
  }

```



```
    },
    "Maceio": {
      "exemplarCity": "מסיאיו"
    },
    "Managua": {
      "exemplarCity": "מנגואה"
    },
    "Manaus": {
      "exemplarCity": "מנאוס"
    },
    "Marigot": {
      "exemplarCity": "מריגו"
    },
    "Martinique": {
      "exemplarCity": "מרטיניק"
    },
    "Matamoros": {
      "exemplarCity": "מטמורוס"
    },
    "Mazatlan": {
      "exemplarCity": "מזטלן"
    },
    "Mendoza": {
      "exemplarCity": "מנדוזזה"
    },
    "Menominee": {
      "exemplarCity": "מנומיני"
    },
    "Merida": {
      "exemplarCity": "מרידה"
    },
    "Metlakatla": {
      "exemplarCity": "מטלקטלה"
    },
    "Mexico_City": {
      "exemplarCity": "מקסיקו סיטי"
    },
    "Miquelon": {
      "exemplarCity": "מיקלון"
    },
    "Moncton": {
      "exemplarCity": "מונקטון"
    },
    "Monterrey": {
      "exemplarCity": "מונטריי"
    },
    "Montevideo": {
      "exemplarCity": "מונטווידאו"
    },
    "Montserrat": {
      "exemplarCity": "מונטראט"
    },
    "Nassau": {
      "exemplarCity": "נסאו"
    },
    "New_York": {
      "exemplarCity": "ניו יורק"
    }
  }
```

```
    },
    "Nipigon": {
      "exemplarCity": "ניפיגון"
    },
    "Nome": {
      "exemplarCity": "נום"
    },
    "Noronha": {
      "exemplarCity": "נורוניה"
    },
    "North_Dakota": {
      "Beulah": {
        "exemplarCity": "ביולה, צפון דקוטה"
      },
      "New_Salem": {
        "exemplarCity": "ניו סילם, צפון דקוטה"
      },
      "Center": {
        "exemplarCity": "סנטר, צפון דקוטה"
      }
    },
    "Ojinaga": {
      "exemplarCity": "אוג'ינאגה"
    },
    "Panama": {
      "exemplarCity": "פנמה"
    },
    "Pangnirtung": {
      "exemplarCity": "פנגנירטונג"
    },
    "Paramaribo": {
      "exemplarCity": "פרמריבו"
    },
    "Phoenix": {
      "exemplarCity": "פיניקס"
    },
    "Port-au-Prince": {
      "exemplarCity": "פורט או פראנס"
    },
    "Port_of_Spain": {
      "exemplarCity": "פורט אוף ספייין"
    },
    "Porto_Velho": {
      "exemplarCity": "פורטו וליו"
    },
    "Puerto_Rico": {
      "exemplarCity": "פוארטו ריקו"
    },
    "Rainy_River": {
      "exemplarCity": "רייני ריבר"
    },
    "Rankin_Inlet": {
      "exemplarCity": "רנקין אינלט"
    },
    "Recife": {
      "exemplarCity": "רסיפה"
    },
  },
```

```
"Regina": {
  "exemplarCity": "רג'ינה"
},
"Resolute": {
  "exemplarCity": "רזולוט"
},
"Rio_Branco": {
  "exemplarCity": "ריו ברנקו"
},
"Santa_Isabel": {
  "exemplarCity": "סנטה איסבל"
},
"Santarem": {
  "exemplarCity": "סנטרם"
},
"Santiago": {
  "exemplarCity": "סנטיאגו"
},
"Santo_Domingo": {
  "exemplarCity": "סנטו דומינגו"
},
"Sao_Paulo": {
  "exemplarCity": "סאו פאולו"
},
"Scoresbysund": {
  "exemplarCity": "סקורסביסונד"
},
"Sitka": {
  "exemplarCity": "סיטקה"
},
"St_Barthelemy": {
  "exemplarCity": "סנט ברתלמי"
},
"St_Johns": {
  "exemplarCity": "סנט ג'ונס"
},
"St_Kitts": {
  "exemplarCity": "סנט קיטס"
},
"St_Lucia": {
  "exemplarCity": "סנט לוסיה"
},
"St_Thomas": {
  "exemplarCity": "סנט תומאס"
},
"St_Vincent": {
  "exemplarCity": "סנט וינסנט"
},
"Swift_Current": {
  "exemplarCity": "סוויפט קרנט"
},
"Tegucigalpa": {
  "exemplarCity": "טגוסיגלפה"
},
"Thule": {
  "exemplarCity": "תולה"
},
},
```

```

    "Thunder_Bay": {
      "exemplarCity": "ת'אנדר ביי"
    },
    "Tijuana": {
      "exemplarCity": "טיחואנה"
    },
    "Toronto": {
      "exemplarCity": "טורונטו"
    },
    "Tortola": {
      "exemplarCity": "טורטולה"
    },
    "Vancouver": {
      "exemplarCity": "ונקובר"
    },
    "Whitehorse": {
      "exemplarCity": "ווייטהורס"
    },
    "Winnipeg": {
      "exemplarCity": "וויניפג"
    },
    "Yakutat": {
      "exemplarCity": "יקוטאט"
    },
    "Yellowknife": {
      "exemplarCity": "ילונייף"
    }
  },
  "Atlantic": {
    "Azores": {
      "exemplarCity": "האיים האזוריים"
    },
    "Bermuda": {
      "exemplarCity": "ברמודה"
    },
    "Canary": {
      "exemplarCity": "האיים הקנריים"
    },
    "Cape_Verde": {
      "exemplarCity": "כף ורדה"
    },
    "Faeroe": {
      "exemplarCity": "פארו"
    },
    "Madeira": {
      "exemplarCity": "מדיירה"
    },
    "Reykjavik": {
      "exemplarCity": "רייקיאוויק"
    },
    "South_Georgia": {
      "exemplarCity": "דרום ג'ורג'יה"
    },
    "St_Helena": {
      "exemplarCity": "סנט הלנה"
    },
    "Stanley": {

```

```

        "exemplarCity": "סטנלי"
    },
    },
    "Europe": {
        "Amsterdam": {
            "exemplarCity": "אמסטרדם"
        },
        "Andorra": {
            "exemplarCity": "אנדורה"
        },
        "Astrakhan": {
            "exemplarCity": "אסטרן"
        },
        "Athens": {
            "exemplarCity": "אתונה"
        },
        "Belgrade": {
            "exemplarCity": "בלגרד"
        },
        "Berlin": {
            "exemplarCity": "ברלין"
        },
        "Bratislava": {
            "exemplarCity": "ברטיסלבה"
        },
        "Brussels": {
            "exemplarCity": "בריסל"
        },
        "Bucharest": {
            "exemplarCity": "בוקרשט"
        },
        "Budapest": {
            "exemplarCity": "בודפשט"
        },
        "Busingen": {
            "exemplarCity": "ביזינגן"
        },
        "Chisinau": {
            "exemplarCity": "קיישינב"
        },
        "Copenhagen": {
            "exemplarCity": "קופנהגן"
        },
        "Dublin": {
            "long": {
                "daylight": "שעון קיץ אירלנד"
            },
            "exemplarCity": "דבלין"
        },
        "Gibraltar": {
            "exemplarCity": "גיברלטר"
        },
        "Guernsey": {
            "exemplarCity": "גרנזי"
        },
        "Helsinki": {
            "exemplarCity": "הלסינקי"
        }
    }
}

```

```
    },
    "Isle_of_Man": {
      "exemplarCity": "האי מאן"
    },
    "Istanbul": {
      "exemplarCity": "איסטנבול"
    },
    "Jersey": {
      "exemplarCity": "ג'רזי"
    },
    "Kaliningrad": {
      "exemplarCity": "קלינינגרד"
    },
    "Kiev": {
      "exemplarCity": "קייב"
    },
    "Kirov": {
      "exemplarCity": "קירוב"
    },
    "Lisbon": {
      "exemplarCity": "ליסבון"
    },
    "Ljubljana": {
      "exemplarCity": "לובליאנה"
    },
    "London": {
      "long": {
        "daylight": "שעון קיץ בריטניה"
      },
      "exemplarCity": "לונדון"
    },
    "Luxembourg": {
      "exemplarCity": "לוקסמבורג"
    },
    "Madrid": {
      "exemplarCity": "מדריד"
    },
    "Malta": {
      "exemplarCity": "מלטה"
    },
    "Mariehamn": {
      "exemplarCity": "מרייהאמן"
    },
    "Minsk": {
      "exemplarCity": "מינסק"
    },
    "Monaco": {
      "exemplarCity": "מונקו"
    },
    "Moscow": {
      "exemplarCity": "מוסקבה"
    },
    "Oslo": {
      "exemplarCity": "אוסלו"
    },
    "Paris": {
      "exemplarCity": "פריז"
```

```
    },  
    "Podgorica": {  
      "exemplarCity": "פודגוריצה"  
    },  
    "Prague": {  
      "exemplarCity": "פראג"  
    },  
    "Riga": {  
      "exemplarCity": "ריגה"  
    },  
    "Rome": {  
      "exemplarCity": "רומא"  
    },  
    "Samara": {  
      "exemplarCity": "סמרה"  
    },  
    "San_Marino": {  
      "exemplarCity": "סן מרינו"  
    },  
    "Sarajevo": {  
      "exemplarCity": "סרייבו"  
    },  
    "Simferopol": {  
      "exemplarCity": "סימפרופול"  
    },  
    "Skopje": {  
      "exemplarCity": "סקופיה"  
    },  
    "Sofia": {  
      "exemplarCity": "סופיה"  
    },  
    "Stockholm": {  
      "exemplarCity": "סטוקהולם"  
    },  
    "Tallinn": {  
      "exemplarCity": "טאלין"  
    },  
    "Tirane": {  
      "exemplarCity": "טירנה"  
    },  
    "Ulyanovsk": {  
      "exemplarCity": "אוליאנובסק"  
    },  
    "Uzhgorod": {  
      "exemplarCity": "אוז'הורוד"  
    },  
    "Vaduz": {  
      "exemplarCity": "ואדוץ"  
    },  
    "Vatican": {  
      "exemplarCity": "הוותיקן"  
    },  
    "Vienna": {  
      "exemplarCity": "וינה"  
    },  
    "Vilnius": {  
      "exemplarCity": "וילנה"
```

```
    },  
    "Volgograd": {  
      "exemplarCity": "וולגוגרד"  
    },  
    "Warsaw": {  
      "exemplarCity": "ורשה"  
    },  
    "Zagreb": {  
      "exemplarCity": "זאגרב"  
    },  
    "Zaporozhye": {  
      "exemplarCity": "זפורוז'יה"  
    },  
    "Zurich": {  
      "exemplarCity": "ציריך"  
    }  
  },  
  "Africa": {  
    "Abidjan": {  
      "exemplarCity": "אביג'אן"  
    },  
    "Accra": {  
      "exemplarCity": "אקרה"  
    },  
    "Addis_Ababa": {  
      "exemplarCity": "אדיס אבבה"  
    },  
    "Algiers": {  
      "exemplarCity": "אלג'יר"  
    },  
    "Asmera": {  
      "exemplarCity": "אסמרה"  
    },  
    "Bamako": {  
      "exemplarCity": "במאקו"  
    },  
    "Bangui": {  
      "exemplarCity": "בנגואי"  
    },  
    "Banjul": {  
      "exemplarCity": "בנג'ול"  
    },  
    "Bissau": {  
      "exemplarCity": "ביסאו"  
    },  
    "Blantyre": {  
      "exemplarCity": "בלנטיר"  
    },  
    "Brazzaville": {  
      "exemplarCity": "ברזוויל"  
    },  
    "Bujumbura": {  
      "exemplarCity": "בוג'ומבורה"  
    },  
    "Cairo": {  
      "exemplarCity": "קהיר"  
    },  
  },  
}
```



```
"Casablanca": {
  "exemplarCity": "קזבלנקה"
},
"Ceuta": {
  "exemplarCity": "סאוטה"
},
"Conakry": {
  "exemplarCity": "קונאקרי"
},
"Dakar": {
  "exemplarCity": "דקאר"
},
"Dar_es_Salaam": {
  "exemplarCity": "דאר א-סלאם"
},
"Djibouti": {
  "exemplarCity": "ג'יבוטי"
},
"Douala": {
  "exemplarCity": "דואלה"
},
"El_Aaiun": {
  "exemplarCity": "אל עיון"
},
"Freetown": {
  "exemplarCity": "פריטאון"
},
"Gaborone": {
  "exemplarCity": "גבורונה"
},
"Harare": {
  "exemplarCity": "הררה"
},
"Johannesburg": {
  "exemplarCity": "יוהנסבורג"
},
"Juba": {
  "exemplarCity": "ג'ובה"
},
"Kampala": {
  "exemplarCity": "קמפלה"
},
"Khartoum": {
  "exemplarCity": "חרטום"
},
"Kigali": {
  "exemplarCity": "קיגלי"
},
"Kinshasa": {
  "exemplarCity": "קינשה"
},
"Lagos": {
  "exemplarCity": "לגוס"
},
"Libreville": {
  "exemplarCity": "ליברוויל"
},
},
```

```
"Lome": {
  "exemplarCity": "לומה"
},
"Luanda": {
  "exemplarCity": "לואנדה"
},
"Lubumbashi": {
  "exemplarCity": "לובומבאשי"
},
"Lusaka": {
  "exemplarCity": "לוסקה"
},
"Malabo": {
  "exemplarCity": "מלבו"
},
"Maputo": {
  "exemplarCity": "מאפוטו"
},
"Maseru": {
  "exemplarCity": "מסרו"
},
"Mbabane": {
  "exemplarCity": "אמבאבאנה"
},
"Mogadishu": {
  "exemplarCity": "מוגדישו"
},
"Monrovia": {
  "exemplarCity": "מונרוביה"
},
"Nairobi": {
  "exemplarCity": "ניירובי"
},
"Ndjamena": {
  "exemplarCity": "נג'מנה"
},
"Niamey": {
  "exemplarCity": "ניאמי"
},
"Nouakchott": {
  "exemplarCity": "נואקצ'וט"
},
"Ouagadougou": {
  "exemplarCity": "וואגאדוגו"
},
"Porto-Novo": {
  "exemplarCity": "פורטו נובו"
},
"Sao_Tome": {
  "exemplarCity": "סאו טומה"
},
"Tripoli": {
  "exemplarCity": "טריפולי"
},
"Tunis": {
  "exemplarCity": "תוניס"
},
},
```

```
"Windhoek": {
  "exemplarCity": "ווינדהוק"
},
"Asia": {
  "Aden": {
    "exemplarCity": "עדן"
  },
  "Almaty": {
    "exemplarCity": "אלמאטי"
  },
  "Amman": {
    "exemplarCity": "עמאן"
  },
  "Anadyr": {
    "exemplarCity": "אנדיר"
  },
  "Aqtau": {
    "exemplarCity": "אקטאו"
  },
  "Aqtobe": {
    "exemplarCity": "אקטובה"
  },
  "Ashgabat": {
    "exemplarCity": "אשגבט"
  },
  "Baghdad": {
    "exemplarCity": "בגדד"
  },
  "Bahrain": {
    "exemplarCity": "בחריין"
  },
  "Baku": {
    "exemplarCity": "באקו"
  },
  "Bangkok": {
    "exemplarCity": "בנגקוק"
  },
  "Barnaul": {
    "exemplarCity": "ברנאול"
  },
  "Beirut": {
    "exemplarCity": "ביירות"
  },
  "Bishkek": {
    "exemplarCity": "בישקק"
  },
  "Brunei": {
    "exemplarCity": "ברוניי"
  },
  "Calcutta": {
    "exemplarCity": "קולקטה"
  },
  "Chita": {
    "exemplarCity": "צ'יטה"
  },
  "Choibalsan": {
```

```
    "exemplarCity": "צ'ויבלסן"  
  },  
  "Colombo": {  
    "exemplarCity": "קולומבו"  
  },  
  "Damascus": {  
    "exemplarCity": "דמשק"  
  },  
  "Dhaka": {  
    "exemplarCity": "דאקה"  
  },  
  "Dili": {  
    "exemplarCity": "דילי"  
  },  
  "Dubai": {  
    "exemplarCity": "דובאי"  
  },  
  "Dushanbe": {  
    "exemplarCity": "דושנבה"  
  },  
  "Gaza": {  
    "exemplarCity": "עזה"  
  },  
  "Hebron": {  
    "exemplarCity": "חברון"  
  },  
  "Hong_Kong": {  
    "exemplarCity": "הונג קונג"  
  },  
  "Hovd": {  
    "exemplarCity": "חובד"  
  },  
  "Irkutsk": {  
    "exemplarCity": "אירקוטסק"  
  },  
  "Jakarta": {  
    "exemplarCity": "ג'קרטה"  
  },  
  "Jayapura": {  
    "exemplarCity": "ג'איאפורה"  
  },  
  "Jerusalem": {  
    "exemplarCity": "ירושלים"  
  },  
  "Kabul": {  
    "exemplarCity": "קאבול"  
  },  
  "Kamchatka": {  
    "exemplarCity": "קמצ'טקה"  
  },  
  "Karachi": {  
    "exemplarCity": "קראצ'י"  
  },  
  "Katmandu": {  
    "exemplarCity": "קטמנדו"  
  },  
  "Khandyga": {
```

```

    "exemplarCity": "חנדיגה"
  },
  "Krasnoyarsk": {
    "exemplarCity": "קרסנויארסק"
  },
  "Kuala_Lumpur": {
    "exemplarCity": "קואלה לומפור"
  },
  "Kuching": {
    "exemplarCity": "קוצ'ינג"
  },
  "Kuwait": {
    "exemplarCity": "כווית"
  },
  "Macau": {
    "exemplarCity": "מקאו"
  },
  "Magadan": {
    "exemplarCity": "מגדן"
  },
  "Makassar": {
    "exemplarCity": "מאקאסאר"
  },
  "Manila": {
    "exemplarCity": "מנילה"
  },
  "Muscat": {
    "exemplarCity": "מוסקט"
  },
  "Nicosia": {
    "exemplarCity": "ניקוסיה"
  },
  "Novokuznetsk": {
    "exemplarCity": "נובוקוזנטסק"
  },
  "Novosibirsk": {
    "exemplarCity": "נובוסיבירסק"
  },
  "Omsk": {
    "exemplarCity": "אומסק"
  },
  "Oral": {
    "exemplarCity": "אורל"
  },
  "Phnom_Penh": {
    "exemplarCity": "פנום פן"
  },
  "Pontianak": {
    "exemplarCity": "פונטיאנק"
  },
  "Pyongyang": {
    "exemplarCity": "פיונגיאנג"
  },
  "Qatar": {
    "exemplarCity": "קטאר"
  },
  "Qyzylorda": {

```

```

    "exemplarCity": "קִיזִילֹרְדֶה"
  },
  "Rangoon": {
    "exemplarCity": "רֶנְגּוּן"
  },
  "Riyadh": {
    "exemplarCity": "רִיאַד"
  },
  "Saigon": {
    "exemplarCity": "הוֹ צִי מִין סִיטִי"
  },
  "Sakhalin": {
    "exemplarCity": "סַחְלִין"
  },
  "Samarkand": {
    "exemplarCity": "סַמַּרְקַנְד"
  },
  "Seoul": {
    "exemplarCity": "סֵאוּל"
  },
  "Shanghai": {
    "exemplarCity": "שַׁנְחַאִי"
  },
  "Singapore": {
    "exemplarCity": "סִינְגַּפּוּר"
  },
  "Srednekolymsk": {
    "exemplarCity": "סֶרְדְנִיקוֹלִימְסֵק"
  },
  "Taipei": {
    "exemplarCity": "טַאִיפֵי"
  },
  "Tashkent": {
    "exemplarCity": "טַשְׁקֶנְט"
  },
  "Tbilisi": {
    "exemplarCity": "טְבִילִיסִי"
  },
  "Tehran": {
    "exemplarCity": "טֶהֶרַן"
  },
  "Thimphu": {
    "exemplarCity": "טְהִימְפּוּ"
  },
  "Tokyo": {
    "exemplarCity": "טּוֹקִיו"
  },
  "Tomsk": {
    "exemplarCity": "טוֹמְסֵק"
  },
  "Ulaanbaatar": {
    "exemplarCity": "אוּלַאנְבַּטַּאר"
  },
  "Urumqi": {
    "exemplarCity": "אוּרוּמְקִי"
  },
  "Ust-Nera": {

```

```

        "exemplarCity": "אוסט-נרה"
    },
    "Vientiane": {
        "exemplarCity": "האנוי"
    },
    "Vladivostok": {
        "exemplarCity": "ולדיוווסטוק"
    },
    "Yakutsk": {
        "exemplarCity": "יקוטסק"
    },
    "Yekaterinburg": {
        "exemplarCity": "יקטרינבורג"
    },
    "Yerevan": {
        "exemplarCity": "ירוואן"
    }
},
"Indian": {
    "Antananarivo": {
        "exemplarCity": "אנטננריבו"
    },
    "Chagos": {
        "exemplarCity": "צ'אגוס"
    },
    "Christmas": {
        "exemplarCity": "האי כריסטמס"
    },
    "Cocos": {
        "exemplarCity": "קוקוס"
    },
    "Comoro": {
        "exemplarCity": "קומורו"
    },
    "Kerguelen": {
        "exemplarCity": "קרגוולן"
    },
    "Mahe": {
        "exemplarCity": "מהא"
    },
    "Maldives": {
        "exemplarCity": "האיים המלדיביים"
    },
    "Mauritius": {
        "exemplarCity": "מאוריציוס"
    },
    "Mayotte": {
        "exemplarCity": "מאיוט"
    },
    "Reunion": {
        "exemplarCity": "ראוניון"
    }
},
"Australia": {
    "Adelaide": {
        "exemplarCity": "אדלייד"
    },

```

```
"Brisbane": {
  "exemplarCity": "בריסביין"
},
"Broken_Hill": {
  "exemplarCity": "ברוקן היל"
},
"Currie": {
  "exemplarCity": "קרי"
},
"Darwin": {
  "exemplarCity": "דרווין"
},
"Eucla": {
  "exemplarCity": "יוקלה"
},
"Hobart": {
  "exemplarCity": "הוברט"
},
"Lindeman": {
  "exemplarCity": "לינדמן"
},
"Lord_Howe": {
  "exemplarCity": "אי הלורד האו"
},
"Melbourne": {
  "exemplarCity": "מלבורן"
},
"Perth": {
  "exemplarCity": "פרת'י"
},
"Sydney": {
  "exemplarCity": "סידני"
}
},
"Pacific": {
  "Apia": {
    "exemplarCity": "אפיה"
  },
  "Auckland": {
    "exemplarCity": "אוקלנד"
  },
  "Bougainville": {
    "exemplarCity": "בוגנוויל"
  },
  "Chatham": {
    "exemplarCity": "צ'אטהאם"
  },
  "Easter": {
    "exemplarCity": "אי הפסחא"
  },
  "Efate": {
    "exemplarCity": "אפטא"
  },
  "Enderbury": {
    "exemplarCity": "אנדרבורי"
  },
  "Fakaofu": {
```



```

    "exemplarCity": "פֶּקֶאוּפּוּ"
  },
  "Fiji": {
    "exemplarCity": "פִּיג'י"
  },
  "Funafuti": {
    "exemplarCity": "פּוֹנַפּוּטִי"
  },
  "Galapagos": {
    "exemplarCity": "גַּלפָּאגוֹס"
  },
  "Gambier": {
    "exemplarCity": "אֵי גַמְבִּייה"
  },
  "Guadalcanal": {
    "exemplarCity": "גוּוֹדְלֶקְנָאֵל"
  },
  "Guam": {
    "exemplarCity": "גוּאָם"
  },
  "Honolulu": {
    "exemplarCity": "הוֹנוּלוּלוּ"
  },
  "Johnston": {
    "exemplarCity": "ג'וֹנסְטוֹן"
  },
  "Kiritimati": {
    "exemplarCity": "קִירִיטִימַאטִי"
  },
  "Kosrae": {
    "exemplarCity": "קוֹסְרֵה"
  },
  "Kwajalein": {
    "exemplarCity": "קוּוֹאג'ֵלֵיין"
  },
  "Majuro": {
    "exemplarCity": "מַאג'וּרוּ"
  },
  "Marquesas": {
    "exemplarCity": "אֵי מַרְקִיז"
  },
  "Midway": {
    "exemplarCity": "מִידוּוֵי"
  },
  "Nauru": {
    "exemplarCity": "נַאוּרוּ"
  },
  "Niue": {
    "exemplarCity": "נִיוֵאה"
  },
  "Norfolk": {
    "exemplarCity": "נֹרפֹּק"
  },
  "Noumea": {
    "exemplarCity": "נוּמֵאה"
  },
  "Pago_Pago": {

```

```

        "exemplarCity": "פאגו פאגו"
    },
    "Palau": {
        "exemplarCity": "פלאו"
    },
    "Pitcairn": {
        "exemplarCity": "פיטקרן"
    },
    "Ponape": {
        "exemplarCity": "פונפיי"
    },
    "Port_Moresby": {
        "exemplarCity": "פורט מורסבי"
    },
    "Rarotonga": {
        "exemplarCity": "רארוטונגה"
    },
    "Saipan": {
        "exemplarCity": "סאיפאן"
    },
    "Tahiti": {
        "exemplarCity": "טהיטי"
    },
    "Tarawa": {
        "exemplarCity": "טאראווה"
    },
    "Tongatapu": {
        "exemplarCity": "טונגטאפו"
    },
    "Truk": {
        "exemplarCity": "צ'וק"
    },
    "Wake": {
        "exemplarCity": "וויק"
    },
    "Wallis": {
        "exemplarCity": "ווליס"
    }
},
"Arctic": {
    "Longyearbyen": {
        "exemplarCity": "לונגיירבין"
    }
},
"Antarctica": {
    "Casey": {
        "exemplarCity": "קאסיי"
    },
    "Davis": {
        "exemplarCity": "דיוויס"
    },
    "DumontDUrville": {
        "exemplarCity": "דומון ד'אורוויל"
    },
    "Macquarie": {
        "exemplarCity": "מקרי"
    }
},

```

```
"Mawson": {
  "exemplarCity": "מוסטון"
},
"McMurdo": {
  "exemplarCity": "מק-מרדון"
},
"Palmer": {
  "exemplarCity": "פאלמר"
},
"Rothera": {
  "exemplarCity": "רות'רה"
},
"Syowa": {
  "exemplarCity": "סיוואה"
},
"Troll": {
  "exemplarCity": "טרול"
},
"Vostok": {
  "exemplarCity": "ווסטוק"
}
},
"Etc": {
  "GMT": {
    "exemplarCity": "GMT"
  },
  "GMT1": {
    "exemplarCity": "GMT+1"
  },
  "GMT10": {
    "exemplarCity": "GMT+10"
  },
  "GMT11": {
    "exemplarCity": "GMT+11"
  },
  "GMT12": {
    "exemplarCity": "GMT+12"
  },
  "GMT2": {
    "exemplarCity": "GMT+2"
  },
  "GMT3": {
    "exemplarCity": "GMT+3"
  },
  "GMT4": {
    "exemplarCity": "GMT+4"
  },
  "GMT5": {
    "exemplarCity": "GMT+5"
  },
  "GMT6": {
    "exemplarCity": "GMT+6"
  },
  "GMT7": {
    "exemplarCity": "GMT+7"
  },
  "GMT8": {
```

```
    "exemplarCity": "GMT+8"
  },
  "GMT9": {
    "exemplarCity": "GMT+9"
  },
  "GMT-1": {
    "exemplarCity": "GMT-1"
  },
  "GMT-10": {
    "exemplarCity": "GMT-10"
  },
  "GMT-11": {
    "exemplarCity": "GMT-11"
  },
  "GMT-12": {
    "exemplarCity": "GMT-12"
  },
  "GMT-13": {
    "exemplarCity": "GMT-13"
  },
  "GMT-14": {
    "exemplarCity": "GMT-14"
  },
  "GMT-2": {
    "exemplarCity": "GMT-2"
  },
  "GMT-3": {
    "exemplarCity": "GMT-3"
  },
  "GMT-4": {
    "exemplarCity": "GMT-4"
  },
  "GMT-5": {
    "exemplarCity": "GMT-5"
  },
  "GMT-6": {
    "exemplarCity": "GMT-6"
  },
  "GMT-7": {
    "exemplarCity": "GMT-7"
  },
  "GMT-8": {
    "exemplarCity": "GMT-8"
  },
  "GMT-9": {
    "exemplarCity": "GMT-9"
  },
  "UTC": {
    "long": {
      "standard": "זמן אוניברסלי מתואם"
    },
    "short": {
      "standard": "UTC"
    },
    "exemplarCity": "UTC"
  },
  "Unknown": {
```

```

        "exemplarCity": "עיר לא ידועה"
    }
},
"metazone": {
    "Afghanistan": {
        "long": {
            "standard": "שעון אפגניסטן"
        }
    },
    "Africa_Central": {
        "long": {
            "standard": "שעון מרכז אפריקה"
        }
    },
    "Africa_Eastern": {
        "long": {
            "standard": "שעון מזרח אפריקה"
        }
    },
    "Africa_Southern": {
        "long": {
            "standard": "שעון דרום אפריקה"
        }
    },
    "Africa_Western": {
        "long": {
            "generic": "שעון מערב אפריקה",
            "standard": "שעון מערב אפריקה (חורף)",
            "daylight": "שעון מערב אפריקה (קיץ)"
        }
    },
    "Alaska": {
        "long": {
            "generic": "שעון אלסקה",
            "standard": "שעון אלסקה (חורף)",
            "daylight": "שעון אלסקה (קיץ)"
        }
    },
    "Amazon": {
        "long": {
            "generic": "שעון אמזונס",
            "standard": "שעון אמזונס (חורף)",
            "daylight": "שעון אמזונס (קיץ)"
        }
    },
    "America_Central": {
        "long": {
            "generic": "שעון מרכז ארה"ב",
            "standard": "שעון מרכז ארה"ב (חורף)",
            "daylight": "שעון מרכז ארה"ב (קיץ)"
        }
    },
    "America_Eastern": {
        "long": {
            "generic": "שעון החוף המזרחי",
            "standard": "שעון החוף המזרחי (חורף)",

```

```

        "daylight": "שעון החוף המזרחי (קיץ)"
    },
    },
    "America_Mountain": {
        "long": {
            "generic": "שעון אזור ההרים בארה"ב",
            "standard": "שעון אזור ההרים בארה"ב (חורף)",
            "daylight": "שעון אזור ההרים בארה"ב (קיץ)"
        }
    },
    "America_Pacific": {
        "long": {
            "generic": "שעון מערב ארה"ב",
            "standard": "שעון מערב ארה"ב (חורף)",
            "daylight": "שעון מערב ארה"ב (קיץ)"
        }
    },
    "Anadyr": {
        "long": {
            "generic": "שעון אנדיר",
            "standard": "שעון רגיל אנדיר",
            "daylight": "שעון קיץ אנדיר"
        }
    },
    "Apia": {
        "long": {
            "generic": "שעון אפיה",
            "standard": "שעון אפיה (חורף)",
            "daylight": "שעון אפיה (קיץ)"
        }
    },
    "Arabian": {
        "long": {
            "generic": "שעון חצי האי ערב",
            "standard": "שעון חצי האי ערב (חורף)",
            "daylight": "שעון חצי האי ערב (קיץ)"
        }
    },
    "Argentina": {
        "long": {
            "generic": "שעון ארגנטינה",
            "standard": "שעון ארגנטינה (חורף)",
            "daylight": "שעון ארגנטינה (קיץ)"
        }
    },
    "Argentina_Western": {
        "long": {
            "generic": "שעון מערב ארגנטינה",
            "standard": "שעון מערב ארגנטינה (חורף)",
            "daylight": "שעון מערב ארגנטינה (קיץ)"
        }
    },
    "Armenia": {
        "long": {
            "generic": "שעון ארמניה",
            "standard": "שעון ארמניה (חורף)",
            "daylight": "שעון ארמניה (קיץ)"
        }
    }

```

```

    }
  },
  "Atlantic": {
    "long": {
      "generic": "שעון האוקיינוס האטלנטי",
      "standard": "שעון האוקיינוס האטלנטי (חורף)",
      "daylight": "שעון האוקיינוס האטלנטי (קיץ)"
    }
  },
  "Australia_Central": {
    "long": {
      "generic": "שעון מרכז אוסטרליה",
      "standard": "שעון מרכז אוסטרליה (חורף)",
      "daylight": "שעון מרכז אוסטרליה (קיץ)"
    }
  },
  "Australia_CentralWestern": {
    "long": {
      "generic": "שעון מרכז-מערב אוסטרליה",
      "standard": "שעון מרכז-מערב אוסטרליה (חורף)",
      "daylight": "שעון מרכז-מערב אוסטרליה (קיץ)"
    }
  },
  "Australia_Eastern": {
    "long": {
      "generic": "שעון מזרח אוסטרליה",
      "standard": "שעון מזרח אוסטרליה (חורף)",
      "daylight": "שעון מזרח אוסטרליה (קיץ)"
    }
  },
  "Australia_Western": {
    "long": {
      "generic": "שעון מערב אוסטרליה",
      "standard": "שעון מערב אוסטרליה (חורף)",
      "daylight": "שעון מערב אוסטרליה (קיץ)"
    }
  },
  "Azerbaijan": {
    "long": {
      "generic": "שעון אזרבייג'אן",
      "standard": "שעון אזרבייג'אן (חורף)",
      "daylight": "שעון אזרבייג'אן (קיץ)"
    }
  },
  "Azores": {
    "long": {
      "generic": "שעון האיים האזוריים",
      "standard": "שעון האיים האזוריים (חורף)",
      "daylight": "שעון האיים האזוריים (קיץ)"
    }
  },
  "Bangladesh": {
    "long": {
      "generic": "שעון בנגלדש",
      "standard": "שעון בנגלדש (חורף)",
      "daylight": "שעון בנגלדש (קיץ)"
    }
  }
}

```

```

    },
    "Bhutan": {
      "long": {
        "standard": "שעון בהוטן"
      }
    },
    "Bolivia": {
      "long": {
        "standard": "שעון בוליביה"
      }
    },
    "Brasilia": {
      "long": {
        "generic": "שעון ברזיליה",
        "standard": "שעון ברזיליה (חורף)",
        "daylight": "שעון ברזיליה (קיץ)"
      }
    },
    "Brunei": {
      "long": {
        "standard": "שעון ברוניי דארוסלאם"
      }
    },
    "Cape_Verde": {
      "long": {
        "generic": "שעון כף ורדה",
        "standard": "שעון כף ורדה (חורף)",
        "daylight": "שעון כף ורדה (קיץ)"
      }
    },
    "Chamorro": {
      "long": {
        "standard": "שעון צ'אמורו"
      }
    },
    "Chatham": {
      "long": {
        "generic": "שעון צ'טהאם",
        "standard": "שעון צ'טהאם (חורף)",
        "daylight": "שעון צ'טהאם (קיץ)"
      }
    },
    "Chile": {
      "long": {
        "generic": "שעון צ'ילה",
        "standard": "שעון צ'ילה (חורף)",
        "daylight": "שעון צ'ילה (קיץ)"
      }
    },
    "China": {
      "long": {
        "generic": "שעון סין",
        "standard": "שעון סין (חורף)",
        "daylight": "שעון סין (קיץ)"
      }
    },
    "Choibalsan": {

```



```

    "long": {
      "generic": "שעון צ'ויבלסן",
      "standard": "שעון צ'ויבלסן (חורף)",
      "daylight": "שעון צ'ויבלסן (קיץ)"
    }
  },
  "Christmas": {
    "long": {
      "standard": "שעון האי כריסטמס"
    }
  },
  "Cocos": {
    "long": {
      "standard": "שעון איי קוקוס"
    }
  },
  "Colombia": {
    "long": {
      "generic": "שעון קולומביה",
      "standard": "שעון קולומביה (חורף)",
      "daylight": "שעון קולומביה (קיץ)"
    }
  },
  "Cook": {
    "long": {
      "generic": "שעון איי קוק",
      "standard": "שעון איי קוק (חורף)",
      "daylight": "שעון איי קוק (מחצית הקיץ)"
    }
  },
  "Cuba": {
    "long": {
      "generic": "שעון קובה",
      "standard": "שעון קובה (חורף)",
      "daylight": "שעון קובה (קיץ)"
    }
  },
  "Davis": {
    "long": {
      "standard": "שעון דייוויס"
    }
  },
  "DumontDUrville": {
    "long": {
      "standard": "שעון דומון ד'אורוויל"
    }
  },
  "East_Timor": {
    "long": {
      "standard": "שעון מזרח טימור"
    }
  },
  "Easter": {
    "long": {
      "generic": "שעון אי הפסחא",
      "standard": "שעון אי הפסחא (חורף)",
      "daylight": "שעון אי הפסחא (קיץ)"
    }
  }
}

```

```

    }
  },
  "Ecuador": {
    "long": {
      "standard": "שעון אקוודור"
    }
  },
  "Europe_Central": {
    "long": {
      "generic": "שעון מרכז אירופה",
      "standard": "שעון מרכז אירופה (חורף)",
      "daylight": "שעון מרכז אירופה (קיץ)"
    }
  },
  "Europe_Eastern": {
    "long": {
      "generic": "שעון מזרח אירופה",
      "standard": "שעון מזרח אירופה (חורף)",
      "daylight": "שעון מזרח אירופה (קיץ)"
    }
  },
  "Europe_Further_Eastern": {
    "long": {
      "standard": "שעון מינסק"
    }
  },
  "Europe_Western": {
    "long": {
      "generic": "שעון מערב אירופה",
      "standard": "שעון מערב אירופה (חורף)",
      "daylight": "שעון מערב אירופה (קיץ)"
    }
  },
  "Falkland": {
    "long": {
      "generic": "שעון איי פוקלנד",
      "standard": "שעון איי פוקלנד (חורף)",
      "daylight": "שעון איי פוקלנד (קיץ)"
    }
  },
  "Fiji": {
    "long": {
      "generic": "שעון פיג'י",
      "standard": "שעון פיג'י (חורף)",
      "daylight": "שעון פיג'י (קיץ)"
    }
  },
  "French_Guiana": {
    "long": {
      "standard": "שעון גיאנה הצרפתית"
    }
  },
  "French_Southern": {
    "long": {
      "standard": "שעון הארצות הדרומיות והאנטארקטיות של צרפת"
    }
  },
},

```

```

"Galapagos": {
  "long": {
    "standard": "שעון איי גלאפגוס"
  }
},
"Gambier": {
  "long": {
    "standard": "שעון איי גמבייה"
  }
},
"Georgia": {
  "long": {
    "generic": "שעון גאורגיה",
    "standard": "שעון גאורגיה (חורף)",
    "daylight": "שעון גאורגיה (קיץ)"
  }
},
"Gilbert_Islands": {
  "long": {
    "standard": "שעון איי גילברט"
  }
},
"GMT": {
  "long": {
    "standard": "שעון גריניץ'"
  }
},
"Greenland_Eastern": {
  "long": {
    "generic": "שעון מזרח גרינלנד",
    "standard": "שעון מזרח גרינלנד (חורף)",
    "daylight": "שעון מזרח גרינלנד (קיץ)"
  }
},
"Greenland_Western": {
  "long": {
    "generic": "שעון מערב גרינלנד",
    "standard": "שעון מערב גרינלנד (חורף)",
    "daylight": "שעון מערב גרינלנד (קיץ)"
  }
},
"Gulf": {
  "long": {
    "standard": "שעון מדינות המפרץ"
  }
},
"Guyana": {
  "long": {
    "standard": "שעון גיאנה"
  }
},
"Hawaii_Aleutian": {
  "long": {
    "generic": "שעון האיים האלאוטיים הוואי",
    "standard": "שעון האיים האלאוטיים הוואי (חורף)",
    "daylight": "שעון האיים האלאוטיים הוואי (קיץ)"
  }
}

```

```

    },
    "Hong_Kong": {
      "long": {
        "generic": "שעון הונג קונג",
        "standard": "שעון הונג קונג (חורף)",
        "daylight": "שעון הונג קונג (קיץ)"
      }
    },
    "Hovd": {
      "long": {
        "generic": "שעון חובד",
        "standard": "שעון חובד (חורף)",
        "daylight": "שעון חובד (קיץ)"
      }
    },
    "India": {
      "long": {
        "standard": "שעון הודו"
      }
    },
    "Indian_Ocean": {
      "long": {
        "standard": "שעון האוקיינוס ההודי"
      }
    },
    "Indochina": {
      "long": {
        "standard": "שעון הודו-סין"
      }
    },
    "Indonesia_Central": {
      "long": {
        "standard": "שעון מרכז אינדונזיה"
      }
    },
    "Indonesia_Eastern": {
      "long": {
        "standard": "שעון מזרח אינדונזיה"
      }
    },
    "Indonesia_Western": {
      "long": {
        "standard": "שעון מערב אינדונזיה"
      }
    },
    "Iran": {
      "long": {
        "generic": "שעון איראן",
        "standard": "שעון איראן (חורף)",
        "daylight": "שעון איראן (קיץ)"
      }
    },
    "Irkutsk": {
      "long": {
        "generic": "שעון אירקוטסק",
        "standard": "שעון אירקוטסק (חורף)",
        "daylight": "שעון אירקוטסק (קיץ)"
      }
    }
  }

```

```

    },
    "Israel": {
      "long": {
        "generic": "שעון ישראל",
        "standard": "שעון ישראל (חורף)",
        "daylight": "שעון ישראל (קיץ)"
      }
    },
    "Japan": {
      "long": {
        "generic": "שעון יפן",
        "standard": "שעון יפן (חורף)",
        "daylight": "שעון יפן (קיץ)"
      }
    },
    "Kamchatka": {
      "long": {
        "generic": "שעון פטרופלובסק-קמצ'טסקי",
        "standard": "שעון רגיל פטרופלובסק-קמצ'טסקי",
        "daylight": "שעון קיץ פטרופלובסק-קמצ'טסקי"
      }
    },
    "Kazakhstan_Eastern": {
      "long": {
        "standard": "שעון מזרח קזחסטן"
      }
    },
    "Kazakhstan_Western": {
      "long": {
        "standard": "שעון מערב קזחסטן"
      }
    },
    "Korea": {
      "long": {
        "generic": "שעון קוריאה",
        "standard": "שעון קוריאה (חורף)",
        "daylight": "שעון קוריאה (קיץ)"
      }
    },
    "Kosrae": {
      "long": {
        "standard": "שעון קוסראה"
      }
    },
    "Krasnoyarsk": {
      "long": {
        "generic": "שעון קרסנויארסק",
        "standard": "שעון קרסנויארסק (חורף)",
        "daylight": "שעון קרסנויארסק (קיץ)"
      }
    },
    "Kyrgystan": {
      "long": {
        "standard": "שעון קירגיזסטן"
      }
    }
  },

```

```
"Line_Islands": {
  "long": {
    "standard": "שעון איי ליין"
  }
},
"Lord_Howe": {
  "long": {
    "generic": "שעון אי הלורד האו",
    "standard": "שעון אי הלורד האו (חורף)",
    "daylight": "שעון אי הלורד האו (קיץ)"
  }
},
"Macau": {
  "long": {
    "generic": "שעון מקאו",
    "standard": "שעון חורף מקאו",
    "daylight": "שעון קיץ מקאו"
  }
},
"Macquarie": {
  "long": {
    "standard": "שעון מקווארי"
  }
},
"Magadan": {
  "long": {
    "generic": "שעון מגדן",
    "standard": "שעון מגדן (חורף)",
    "daylight": "שעון מגדן (קיץ)"
  }
},
"Malaysia": {
  "long": {
    "standard": "שעון מלזיה"
  }
},
"Maldives": {
  "long": {
    "standard": "שעון האיים המלדיביים"
  }
},
"Marquesas": {
  "long": {
    "standard": "שעון איי מרקז"
  }
},
"Marshall_Islands": {
  "long": {
    "standard": "שעון איי מרשל"
  }
},
"Mauritius": {
  "long": {
    "generic": "שעון מאוריזיוס",
    "standard": "שעון מאוריזיוס (חורף)",
    "daylight": "שעון מאוריזיוס (קיץ)"
  }
}
```

```

    },
    "Mawson": {
      "long": {
        "standard": "שעון מאוסון"
      }
    },
    "Mexico_Northwest": {
      "long": {
        "generic": "שעון צפון-מערב מקסיקו",
        "standard": "שעון צפון-מערב מקסיקו (חורף)",
        "daylight": "שעון צפון-מערב מקסיקו (קיץ)"
      }
    },
    "Mexico_Pacific": {
      "long": {
        "generic": "שעון מערב מקסיקו",
        "standard": "שעון מערב מקסיקו (חורף)",
        "daylight": "שעון מערב מקסיקו (קיץ)"
      }
    },
    "Mongolia": {
      "long": {
        "generic": "שעון אולן בטור",
        "standard": "שעון אולן בטור (חורף)",
        "daylight": "שעון אולן בטור (קיץ)"
      }
    },
    "Moscow": {
      "long": {
        "generic": "שעון מוסקבה",
        "standard": "שעון מוסקבה (חורף)",
        "daylight": "שעון מוסקבה (קיץ)"
      }
    },
    "Myanmar": {
      "long": {
        "standard": "שעון מיאנמר"
      }
    },
    "Nauru": {
      "long": {
        "standard": "שעון נאורו"
      }
    },
    "Nepal": {
      "long": {
        "standard": "שעון נפאל"
      }
    },
    "New_Caledonia": {
      "long": {
        "generic": "שעון קלדוניה החדשה",
        "standard": "שעון קלדוניה החדשה (חורף)",
        "daylight": "שעון קלדוניה החדשה (קיץ)"
      }
    },
    "New_Zealand": {

```

```

    "long": {
      "generic": "שעון ניו זילנד",
      "standard": "שעון ניו זילנד (חורף)",
      "daylight": "שעון ניו זילנד (קיץ)"
    }
  },
  "Newfoundland": {
    "long": {
      "generic": "שעון ניופאונדלנד",
      "standard": "שעון ניופאונדלנד (חורף)",
      "daylight": "שעון ניופאונדלנד (קיץ)"
    }
  },
  "Niue": {
    "long": {
      "standard": "שעון ניואה"
    }
  },
  "Norfolk": {
    "long": {
      "standard": "שעון האי נורפוק"
    }
  },
  "Noronha": {
    "long": {
      "generic": "שעון פרננדו די נורוניה",
      "standard": "שעון פרננדו די נורוניה (חורף)",
      "daylight": "שעון פרננדו די נורוניה (קיץ)"
    }
  },
  "Novosibirsk": {
    "long": {
      "generic": "שעון נובוסיבירסק",
      "standard": "שעון נובוסיבירסק (חורף)",
      "daylight": "שעון נובוסיבירסק (קיץ)"
    }
  },
  "Omsk": {
    "long": {
      "generic": "שעון אומסק",
      "standard": "שעון אומסק (חורף)",
      "daylight": "שעון אומסק (קיץ)"
    }
  },
  "Pakistan": {
    "long": {
      "generic": "שעון פקיסטן",
      "standard": "שעון פקיסטן (חורף)",
      "daylight": "שעון פקיסטן (קיץ)"
    }
  },
  "Palau": {
    "long": {
      "standard": "שעון פלאו"
    }
  },
  "Papua New Guinea": {

```



```

    "long": {
      "standard": "שעון פפואה גיניאה החדשה"
    }
  },
  "Paraguay": {
    "long": {
      "generic": "שעון פרגוואי",
      "standard": "שעון פרגוואי (חורף)",
      "daylight": "שעון פרגוואי (קיץ)"
    }
  },
  "Peru": {
    "long": {
      "generic": "שעון פרו",
      "standard": "שעון פרו (חורף)",
      "daylight": "שעון פרו (קיץ)"
    }
  },
  "Philippines": {
    "long": {
      "generic": "שעון הפיליפינים",
      "standard": "שעון הפיליפינים (חורף)",
      "daylight": "שעון הפיליפינים (קיץ)"
    }
  },
  "Phoenix_Islands": {
    "long": {
      "standard": "שעון איי פיניקס"
    }
  },
  "Pierre_Miquelon": {
    "long": {
      "generic": "שעון סנט פייר ומיקלון",
      "standard": "שעון סנט פייר ומיקלון (חורף)",
      "daylight": "שעון סנט פייר ומיקלון (קיץ)"
    }
  },
  "Pitcairn": {
    "long": {
      "standard": "שעון פיטקרן"
    }
  },
  "Ponape": {
    "long": {
      "standard": "שעון פונאפי"
    }
  },
  "Pyongyang": {
    "long": {
      "standard": "שעון פיונגיאנג"
    }
  },
  "Reunion": {
    "long": {
      "standard": "שעון ראוניון"
    }
  },
},

```

```
"Rothera": {
  "long": {
    "standard": "שעון רות'רה"
  }
},
"Sakhalin": {
  "long": {
    "generic": "שעון סחלין",
    "standard": "שעון סחלין (חורף)",
    "daylight": "שעון סחלין (קיץ)"
  }
},
"Samara": {
  "long": {
    "generic": "שעון סמרה",
    "standard": "שעון רגיל סמרה",
    "daylight": "שעון קיץ סמרה"
  }
},
"Samoa": {
  "long": {
    "generic": "שעון סמואה",
    "standard": "שעון סמואה (חורף)",
    "daylight": "שעון סמואה (קיץ)"
  }
},
"Seychelles": {
  "long": {
    "standard": "שעון איי סיישל"
  }
},
"Singapore": {
  "long": {
    "standard": "שעון סינגפור"
  }
},
"Solomon": {
  "long": {
    "standard": "שעון איי שלמה"
  }
},
"South_Georgia": {
  "long": {
    "standard": "שעון דרום ג'ורג'יה"
  }
},
"Suriname": {
  "long": {
    "standard": "שעון סורינאם"
  }
},
"Syowa": {
  "long": {
    "standard": "שעון סיווה"
  }
},
"Tahiti": {
```

```

    "long": {
      "standard": "שעון טהיטי"
    }
  },
  "Taipei": {
    "long": {
      "generic": "שעון טאיפיי",
      "standard": "שעון טאיפיי (חורף)",
      "daylight": "שעון טאיפיי (קיץ)"
    }
  },
  "Tajikistan": {
    "long": {
      "standard": "שעון טג'יקיסטן"
    }
  },
  "Tokelau": {
    "long": {
      "standard": "שעון טוקלאו"
    }
  },
  "Tonga": {
    "long": {
      "generic": "שעון טונגה",
      "standard": "שעון טונגה (חורף)",
      "daylight": "שעון טונגה (קיץ)"
    }
  },
  "Truk": {
    "long": {
      "standard": "שעון צ'וק"
    }
  },
  "Turkmenistan": {
    "long": {
      "generic": "שעון טורקמניסטן",
      "standard": "שעון טורקמניסטן (חורף)",
      "daylight": "שעון טורקמניסטן (קיץ)"
    }
  },
  "Tuvalu": {
    "long": {
      "standard": "שעון טובאלו"
    }
  },
  "Uruguay": {
    "long": {
      "generic": "שעון אורוגוואי",
      "standard": "שעון אורוגוואי (חורף)",
      "daylight": "שעון אורוגוואי (קיץ)"
    }
  },
  "Uzbekistan": {
    "long": {
      "generic": "שעון אוזבקיסטן",
      "standard": "שעון אוזבקיסטן (חורף)",
      "daylight": "שעון אוזבקיסטן (קיץ)"
    }
  }
}

```

```
    },
    "Vanuatu": {
      "long": {
        "generic": "שעון ונואטו",
        "standard": "שעון ונואטו (חורף)",
        "daylight": "שעון ונואטו (קיץ)"
      }
    },
    "Venezuela": {
      "long": {
        "standard": "שעון ונצואלה"
      }
    },
    "Vladivostok": {
      "long": {
        "generic": "שעון ולדיווסטוק",
        "standard": "שעון ולדיווסטוק (חורף)",
        "daylight": "שעון ולדיווסטוק (קיץ)"
      }
    },
    "Volgograd": {
      "long": {
        "generic": "שעון וולגוגרד",
        "standard": "שעון וולגוגרד (חורף)",
        "daylight": "שעון וולגוגרד (קיץ)"
      }
    },
    "Vostok": {
      "long": {
        "standard": "שעון ווסטוק"
      }
    },
    "Wake": {
      "long": {
        "standard": "שעון האי וייק"
      }
    },
    "Wallis": {
      "long": {
        "standard": "שעון וואליס ופוטונה"
      }
    },
    "Yakutsk": {
      "long": {
        "generic": "שעון יקוטסק",
        "standard": "שעון יקוטסק (חורף)",
        "daylight": "שעון יקוטסק (קיץ)"
      }
    },
    "Yekaterinburg": {
      "long": {
        "generic": "שעון יקטרינבורג",
        "standard": "שעון יקטרינבורג (חורף)",
        "daylight": "שעון יקטרינבורג (קיץ)"
      }
    }
  }
}
```

```

    }
  }
}
}
}

```

TIMEZONENAMES.JSX

```

{
  "main";
  {
    "he";
    {
      "identity";
      {
        "version";
        {
          "_number";
          "$Revision: 13259 $",
          "_cldrVersion";
          "31.0.1";
        }
        "language";
        "he";
      }
      "dates";
      {
        "timeZoneNames";
        {
          "hourFormat";
          "+HH:mm;-HH:mm",
          "gmtFormat";
          "GMT{0}",
          "gmtZeroFormat";
          "GMT",
          "regionFormat";
          "שעון {0}",
          "regionFormat-type-daylight";
          "רקיץ (0{ שעון",
          "regionFormat-type-standard";
          "רקיץ (0{ שעון",
          "fallbackFormat";
          "{1} ({0}) ",
          "zone";
          {
            "America";
            {
              "Adak";
              {
                "exemplarCity";
                "קאדאק";
              }
              "Anchorage";
              {
                "exemplarCity";

```

```

        "אנקורג'";
    }
    "Anguilla";
    {
        "exemplarCity";
        "אנגוויילה";
    }
    "Antigua";
    {
        "exemplarCity";
        "אנטיגואה";
    }
    "Araguaina";
    {
        "exemplarCity";
        "אראגואינה";
    }
    "Argentina";
    {
        "Rio_Gallegos";
        {
            "exemplarCity";
            "ריו גאליגוס";
        }
        "San_Juan";
        {
            "exemplarCity";
            "סן חואן";
        }
        "Ushuaia";
        {
            "exemplarCity";
            "אושואיה";
        }
        "La_Rioja";
        {
            "exemplarCity";
            "לה ריוחה";
        }
        "San_Luis";
        {
            "exemplarCity";
            "סן לואיס";
        }
        "Salta";
        {
            "exemplarCity";
            "סלטה";
        }
        "Tucuman";
        {
            "exemplarCity";
            "טוקומן";
        }
    }
    "Aruba";
    {

```

```
        "exemplarCity";
        "ארוזה";
    }
    "Asuncion";
    {
        "exemplarCity";
        "אסונסיון";
    }
    "Bahia";
    {
        "exemplarCity";
        "באהיה";
    }
    "Bahia_Banderas";
    {
        "exemplarCity";
        "באהיה בנדרס";
    }
    "Barbados";
    {
        "exemplarCity";
        "ברבדוס";
    }
    "Belem";
    {
        "exemplarCity";
        "בלם";
    }
    "Belize";
    {
        "exemplarCity";
        "בליז";
    }
    "Blanc-Sablon";
    {
        "exemplarCity";
        "בלאן-סבלון";
    }
    "Boa_Vista";
    {
        "exemplarCity";
        "בואה ויסטה";
    }
    "Bogota";
    {
        "exemplarCity";
        "בוגוטה";
    }
    "Boise";
    {
        "exemplarCity";
        "בוויסי";
    }
    "Buenos_Aires";
    {
        "exemplarCity";
        "בואנוס איירס";
    }
```

```
}
"Cambridge_Bay";
{
  "exemplarCity";
  "קיימברידג' ביי";
}
"Campo_Grande";
{
  "exemplarCity";
  "קמפו גרנדה";
}
"Cancun";
{
  "exemplarCity";
  "קנקון";
}
"Caracas";
{
  "exemplarCity";
  "קראקס";
}
"Catamarca";
{
  "exemplarCity";
  "קטמרקה";
}
"Cayenne";
{
  "exemplarCity";
  "קאיייל";
}
"Cayman";
{
  "exemplarCity";
  "קיימן";
}
"Chicago";
{
  "exemplarCity";
  "שיקגו";
}
"Chihuahua";
{
  "exemplarCity";
  "צ'יוואוואה";
}
"Coral_Harbour";
{
  "exemplarCity";
  "אטיקוקל";
}
"Cordoba";
{
  "exemplarCity";
  "קורדובה";
}
"Costa_Rica";
```



```
{
    "exemplarCity";
    "קוטטה ריקה";
}
"Creston";
{
    "exemplarCity";
    "קרסטון";
}
"Cuiaba";
{
    "exemplarCity";
    "קויאבה";
}
"Curacao";
{
    "exemplarCity";
    "קוראסאו";
}
"Danmarkshavn";
{
    "exemplarCity";
    "דנמרקסהוון";
}
"Dawson";
{
    "exemplarCity";
    "דוסון";
}
"Dawson_Creek";
{
    "exemplarCity";
    "דוסון קריק";
}
"Denver";
{
    "exemplarCity";
    "דנוור";
}
"Detroit";
{
    "exemplarCity";
    "דטרויט";
}
"Dominica";
{
    "exemplarCity";
    "דומיניקה";
}
"Edmonton";
{
    "exemplarCity";
    "אדמונטון";
}
"Eirunepe";
{
    "exemplarCity";
```

```
        "אירונפי";
    }
    "El_Salvador";
    {
        "exemplarCity";
        "אל סלבדור";
    }
    "Fort_Nelson";
    {
        "exemplarCity";
        "פורט נלסון";
    }
    "Fortaleza";
    {
        "exemplarCity";
        "פורטאלזה";
    }
    "Glace_Bay";
    {
        "exemplarCity";
        "גלייס ביי";
    }
    "Godthab";
    {
        "exemplarCity";
        "גואוק";
    }
    "Goose_Bay";
    {
        "exemplarCity";
        "גוס ביי";
    }
    "Grand_Turk";
    {
        "exemplarCity";
        "גרנד טורק";
    }
    "Grenada";
    {
        "exemplarCity";
        "גרנדה";
    }
    "Guadeloupe";
    {
        "exemplarCity";
        "גואדלופ";
    }
    "Guatemala";
    {
        "exemplarCity";
        "גואטמלה";
    }
    "Guayaquil";
    {
        "exemplarCity";
        "גואיאקיל";
    }
}
```

```
"Guyana";
{
  "exemplarCity";
  "גיאנה";
}
"Halifax";
{
  "exemplarCity";
  "הליפקס";
}
"Havana";
{
  "exemplarCity";
  "הוואנה";
}
"Hermosillo";
{
  "exemplarCity";
  "הרמוסיו";
}
"Indiana";
{
  "Vincennes";
  {
    "exemplarCity";
    "וינסנס, אינדיאנה";
  }
  "Petersburg";
  {
    "exemplarCity";
    "פיטרסבורג, אינדיאנה";
  }
  "Tell_City";
  {
    "exemplarCity";
    "טל סיטי, אינדיאנה";
  }
  "Knox";
  {
    "exemplarCity";
    "נוקס, אינדיאנה";
  }
  "Winamac";
  {
    "exemplarCity";
    "ויןמאק, אינדיאנה";
  }
  "Marengo";
  {
    "exemplarCity";
    "מרנגו, אינדיאנה";
  }
  "Vevay";
  {
    "exemplarCity";
    "ויוויי, אינדיאנה";
  }
}
```

```
"Indianapolis";
{
    "exemplarCity";
    "אינדיאנפוליס";
}
"Inuvik";
{
    "exemplarCity";
    "אינוויק";
}
"Iqaluit";
{
    "exemplarCity";
    "אינקלוויט";
}
"Jamaica";
{
    "exemplarCity";
    "ג'מייקה";
}
"Jujuy";
{
    "exemplarCity";
    "חוחוני";
}
"Juneau";
{
    "exemplarCity";
    "ג'ונו";
}
"Kentucky";
{
    "Monticello";
    {
        "exemplarCity";
        "מונטיצ'לו, קנטאקי";
    }
}
"Kralendijk";
{
    "exemplarCity";
    "קרלנדייק";
}
"La_Paz";
{
    "exemplarCity";
    "לאה פאז";
}
"Lima";
{
    "exemplarCity";
    "לימה";
}
"Los_Angeles";
{
    "exemplarCity";
```

```
        "לוס אנג'לס";
    }
    "Louisville";
    {
        "exemplarCity";
        "לואיוויל";
    }
    "Lower_Princes";
    {
        "exemplarCity";
        "לואוור פרינסס קוורטר";
    }
    "Maceio";
    {
        "exemplarCity";
        "מסיאו";
    }
    "Managua";
    {
        "exemplarCity";
        "מנגואה";
    }
    "Manaus";
    {
        "exemplarCity";
        "מנאוס";
    }
    "Marigot";
    {
        "exemplarCity";
        "מריגו";
    }
    "Martinique";
    {
        "exemplarCity";
        "מרטיניק";
    }
    "Matamoros";
    {
        "exemplarCity";
        "מטמורוס";
    }
    "Mazatlan";
    {
        "exemplarCity";
        "מזטלן";
    }
    "Mendoza";
    {
        "exemplarCity";
        "מנדוזזה";
    }
    "Menominee";
    {
        "exemplarCity";
        "מנומיני";
    }
}
```

```
"Merida";
{
  "exemplarCity";
  "מרידה";
}
"Metlakatla";
{
  "exemplarCity";
  "מטלקטלה";
}
"Mexico_City";
{
  "exemplarCity";
  "מקסיקו סיטי";
}
"Miquelon";
{
  "exemplarCity";
  "מיקלון";
}
"Moncton";
{
  "exemplarCity";
  "מונקטון";
}
"Monterrey";
{
  "exemplarCity";
  "מונטריי";
}
"Montevideo";
{
  "exemplarCity";
  "מונטווידאו";
}
"Montserrat";
{
  "exemplarCity";
  "מונטראט";
}
"Nassau";
{
  "exemplarCity";
  "נסאו";
}
"New_York";
{
  "exemplarCity";
  "ניו יורק";
}
"Nipigon";
{
  "exemplarCity";
  "ניפיגון";
}
"Nome";
{
```

```
        "exemplarCity";
        "נום";
    }
    "Noronha";
    {
        "exemplarCity";
        "נורוניה";
    }
    "North_Dakota";
    {
        "Beulah";
        {
            "exemplarCity";
            "ביולה, צפון דקוטה";
        }
        "New_Salem";
        {
            "exemplarCity";
            "ניו סיילם, צפון דקוטה";
        }
        "Center";
        {
            "exemplarCity";
            "סנטר, צפון דקוטה";
        }
    }
    "Ojinaga";
    {
        "exemplarCity";
        "אוג'ינאגה";
    }
    "Panama";
    {
        "exemplarCity";
        "פנמה";
    }
    "Pangnirtung";
    {
        "exemplarCity";
        "פנגנירטונג";
    }
    "Paramaribo";
    {
        "exemplarCity";
        "פרמריבו";
    }
    "Phoenix";
    {
        "exemplarCity";
        "פיניקס";
    }
    "Port-au-Prince";
    {
        "exemplarCity";
        "פורט או פראנס";
    }
    "Port_of_Spain";
```

```
{
  "exemplarCity";
  "פורט אוף ספייין";
}
"Porto_Velho";
{
  "exemplarCity";
  "פורטו וליו";
}
"Puerto_Rico";
{
  "exemplarCity";
  "פוארטו ריקו";
}
"Rainy_River";
{
  "exemplarCity";
  "רייני ריבר";
}
"Rankin_Inlet";
{
  "exemplarCity";
  "רנקין אינלט";
}
"Recife";
{
  "exemplarCity";
  "רסיפה";
}
"Regina";
{
  "exemplarCity";
  "רג'ינה";
}
"Resolute";
{
  "exemplarCity";
  "רזולוט";
}
"Rio_Branco";
{
  "exemplarCity";
  "ריו ברנקו";
}
"Santa_Isabel";
{
  "exemplarCity";
  "סנטה איסבל";
}
"Santarem";
{
  "exemplarCity";
  "סנטרם";
}
"Santiago";
{
  "exemplarCity";
```



```

        "סנט יאגו";
    }
    "Santo_Domingo";
    {
        "exemplarCity";
        "סנטו דומינגו";
    }
    "Sao_Paulo";
    {
        "exemplarCity";
        "סאו פאולו";
    }
    "Scoresbysund";
    {
        "exemplarCity";
        "סקורסביסונד";
    }
    "Sitka";
    {
        "exemplarCity";
        "סיטקה";
    }
    "St_Barthelemy";
    {
        "exemplarCity";
        "סנט ברתלמי";
    }
    "St_Johns";
    {
        "exemplarCity";
        "סנט ג'ונס";
    }
    "St_Kitts";
    {
        "exemplarCity";
        "סנט קיטס";
    }
    "St_Lucia";
    {
        "exemplarCity";
        "סנט לוסיה";
    }
    "St_Thomas";
    {
        "exemplarCity";
        "סנט תומאס";
    }
    "St_Vincent";
    {
        "exemplarCity";
        "סנט וינסנט";
    }
    "Swift_Current";
    {
        "exemplarCity";
        "סוויפט קרנט";
    }
}

```

```
"Tegucigalpa";
{
  "exemplarCity";
  "טגוסִיגלפּה";
}
"Thule";
{
  "exemplarCity";
  "תּוּלֶה";
}
"Thunder_Bay";
{
  "exemplarCity";
  "ת'אנדר ביי";
}
"Tijuana";
{
  "exemplarCity";
  "טיחואנָה";
}
"Toronto";
{
  "exemplarCity";
  "טורונטו";
}
"Tortola";
{
  "exemplarCity";
  "טורטוּלֶה";
}
"Vancouver";
{
  "exemplarCity";
  "וונקובר";
}
"Whitehorse";
{
  "exemplarCity";
  "ווייטהורס";
}
"Winnipeg";
{
  "exemplarCity";
  "וויניפּג";
}
"Yakutat";
{
  "exemplarCity";
  "יקוטאט";
}
"Yellowknife";
{
  "exemplarCity";
  "יילוניף";
}
}
"Atlantic";
```

```
{
  "Azores";
  {
    "exemplarCity";
    "האיים האזוריים";
  }
  "Bermuda";
  {
    "exemplarCity";
    "ברמודה";
  }
  "Canary";
  {
    "exemplarCity";
    "האיים הקנריים";
  }
  "Cape_Verde";
  {
    "exemplarCity";
    "כף ורדה";
  }
  "Faeroe";
  {
    "exemplarCity";
    "פארו";
  }
  "Madeira";
  {
    "exemplarCity";
    "מדירה";
  }
  "Reykjavik";
  {
    "exemplarCity";
    "רייקיאויק";
  }
  "South_Georgia";
  {
    "exemplarCity";
    "דרום ג'ורג'יה";
  }
  "St_Helena";
  {
    "exemplarCity";
    "סנט הלנה";
  }
  "Stanley";
  {
    "exemplarCity";
    "סטנלי";
  }
}
"Europe";
{
  "Amsterdam";
  {
    "exemplarCity";
```

```
        "אמסטרדם";
    }
    "Andorra";
    {
        "exemplarCity";
        "אנדורה";
    }
    "Astrakhan";
    {
        "exemplarCity";
        "אסטרחן";
    }
    "Athens";
    {
        "exemplarCity";
        "אתונה";
    }
    "Belgrade";
    {
        "exemplarCity";
        "בלגרד";
    }
    "Berlin";
    {
        "exemplarCity";
        "ברלין";
    }
    "Bratislava";
    {
        "exemplarCity";
        "ברטיסלבה";
    }
    "Brussels";
    {
        "exemplarCity";
        "בריסל";
    }
    "Bucharest";
    {
        "exemplarCity";
        "בוקרשט";
    }
    "Budapest";
    {
        "exemplarCity";
        "בודפשט";
    }
    "Busingen";
    {
        "exemplarCity";
        "ביזינגן";
    }
    "Chisinau";
    {
        "exemplarCity";
        "קיישוב";
    }
}
```

```
"Copenhagen";
{
  "exemplarCity";
  "קופנהגן";
}
"Dublin";
{
  "long";
  {
    "daylight";
    "שעון קיץ אירלנד";
  }
  "exemplarCity";
  "דבלין";
}
"Gibraltar";
{
  "exemplarCity";
  "גיברלטר";
}
"Guernsey";
{
  "exemplarCity";
  "גרנזי";
}
"Helsinki";
{
  "exemplarCity";
  "הלסינקי";
}
"Isle_of_Man";
{
  "exemplarCity";
  "האי מאן";
}
"Istanbul";
{
  "exemplarCity";
  "איסטנבול";
}
"Jersey";
{
  "exemplarCity";
  "ג'רזי";
}
"Kaliningrad";
{
  "exemplarCity";
  "קלינינגרד";
}
"Kiev";
{
  "exemplarCity";
  "קייב";
}
"Kirov";
{
```

```
        "exemplarCity";
        "קירוב";
    }
    "Lisbon";
    {
        "exemplarCity";
        "ליסבון";
    }
    "Ljubljana";
    {
        "exemplarCity";
        "לובליאנה";
    }
    "London";
    {
        "long";
        {
            "daylight";
            "שעון קיץ בריטניה";
        }
        "exemplarCity";
        "לונדון";
    }
    "Luxembourg";
    {
        "exemplarCity";
        "לוקסמבורג";
    }
    "Madrid";
    {
        "exemplarCity";
        "מדריד";
    }
    "Malta";
    {
        "exemplarCity";
        "מלטה";
    }
    "Mariehamn";
    {
        "exemplarCity";
        "מרייהאמן";
    }
    "Minsk";
    {
        "exemplarCity";
        "מינסק";
    }
    "Monaco";
    {
        "exemplarCity";
        "מונקו";
    }
    "Moscow";
    {
        "exemplarCity";
        "מוסקבה";
    }
```

```
}
"Oslo";
{
  "exemplarCity";
  "אוסלו";
}
"Paris";
{
  "exemplarCity";
  "פריז";
}
"Podgorica";
{
  "exemplarCity";
  "פודגוריצה";
}
"Prague";
{
  "exemplarCity";
  "פראג";
}
"Riga";
{
  "exemplarCity";
  "ריגה";
}
"Rome";
{
  "exemplarCity";
  "רומא";
}
"Samara";
{
  "exemplarCity";
  "סמרה";
}
"San_Marino";
{
  "exemplarCity";
  "סן מרינו";
}
"Sarajevo";
{
  "exemplarCity";
  "סרייבו";
}
"Simferopol";
{
  "exemplarCity";
  "סימפרופול";
}
"Skopje";
{
  "exemplarCity";
  "סקופיה";
}
"Sofia";
```

```
{
  "exemplarCity";
  "סופיה";
}
"Stockholm";
{
  "exemplarCity";
  "שטוקהולם";
}
"Tallinn";
{
  "exemplarCity";
  "טאלין";
}
"Tirane";
{
  "exemplarCity";
  "טירנה";
}
"Ulyanovsk";
{
  "exemplarCity";
  "אוליאנובסק";
}
"Uzhgorod";
{
  "exemplarCity";
  "אוז'הורוד";
}
"Vaduz";
{
  "exemplarCity";
  "ואדוץ";
}
"Vatican";
{
  "exemplarCity";
  "הוותיקן";
}
"Vienna";
{
  "exemplarCity";
  "וינה";
}
"Vilnius";
{
  "exemplarCity";
  "וילנה";
}
"Volgograd";
{
  "exemplarCity";
  "וולגוגרד";
}
"Warsaw";
{
  "exemplarCity";
```



```
        "ורשה";
    }
    "Zagreb";
    {
        "exemplarCity";
        "זאגרב";
    }
    "Zaporozhye";
    {
        "exemplarCity";
        "זפורוז'יה";
    }
    "Zurich";
    {
        "exemplarCity";
        "ציריך";
    }
}
"Africa";
{
    "Abidjan";
    {
        "exemplarCity";
        "אביג'אן";
    }
    "Accra";
    {
        "exemplarCity";
        "אקרה";
    }
    "Addis_Ababa";
    {
        "exemplarCity";
        "אדיס אבבה";
    }
    "Algiers";
    {
        "exemplarCity";
        "אלג'יר";
    }
    "Asmera";
    {
        "exemplarCity";
        "אסמרה";
    }
    "Bamako";
    {
        "exemplarCity";
        "במאקו";
    }
    "Bangui";
    {
        "exemplarCity";
        "בנגווי";
    }
    "Banjul";
    {
```

```
        "exemplarCity";
        "בונג'ול";
    }
    "Bissau";
    {
        "exemplarCity";
        "ביסאו";
    }
    "Blantyre";
    {
        "exemplarCity";
        "בלנטיר";
    }
    "Brazzaville";
    {
        "exemplarCity";
        "ברזוויל";
    }
    "Bujumbura";
    {
        "exemplarCity";
        "בוג'ומבורה";
    }
    "Cairo";
    {
        "exemplarCity";
        "קהיר";
    }
    "Casablanca";
    {
        "exemplarCity";
        "קזבלנקה";
    }
    "Ceuta";
    {
        "exemplarCity";
        "סאוטה";
    }
    "Conakry";
    {
        "exemplarCity";
        "קונאקרי";
    }
    "Dakar";
    {
        "exemplarCity";
        "דקאר";
    }
    "Dar_es_Salaam";
    {
        "exemplarCity";
        "דאר א-סלאם";
    }
    "Djibouti";
    {
        "exemplarCity";
        "גיבוטי";
    }
```

```
}
"Douala";
{
  "exemplarCity";
  "דואלה";
}
"El_Aaiun";
{
  "exemplarCity";
  "אל עיון";
}
"Freetown";
{
  "exemplarCity";
  "פריטאון";
}
"Gaborone";
{
  "exemplarCity";
  "גבורונה";
}
"Harare";
{
  "exemplarCity";
  "הרארה";
}
"Johannesburg";
{
  "exemplarCity";
  "יוהנסבורג";
}
"Juba";
{
  "exemplarCity";
  "ג'ובה";
}
"Kampala";
{
  "exemplarCity";
  "קמפלה";
}
"Khartoum";
{
  "exemplarCity";
  "חרטום";
}
"Kigali";
{
  "exemplarCity";
  "קיגלי";
}
"Kinshasa";
{
  "exemplarCity";
  "קינשה";
}
"Lagos";
```

```
{
  "exemplarCity";
  "לגווס";
}
"Libreville";
{
  "exemplarCity";
  "ליברוויל";
}
"Lome";
{
  "exemplarCity";
  "לומה";
}
"Luanda";
{
  "exemplarCity";
  "לואנדה";
}
"Lubumbashi";
{
  "exemplarCity";
  "לובומבאשי";
}
"Lusaka";
{
  "exemplarCity";
  "לוסקה";
}
"Malabo";
{
  "exemplarCity";
  "מלבו";
}
"Maputo";
{
  "exemplarCity";
  "מאפוטו";
}
"Maseru";
{
  "exemplarCity";
  "מסרו";
}
"Mbabane";
{
  "exemplarCity";
  "אמבאבאנה";
}
"Mogadishu";
{
  "exemplarCity";
  "מוגדישו";
}
"Monrovia";
{
  "exemplarCity";
```

```

        "מונרוביה";
    }
    "Nairobi";
    {
        "exemplarCity";
        "ניירובי";
    }
    "Ndjamena";
    {
        "exemplarCity";
        "נג'מנה";
    }
    "Niamey";
    {
        "exemplarCity";
        "ניאמי";
    }
    "Nouakchott";
    {
        "exemplarCity";
        "נואקצ'וט";
    }
    "Ouagadougou";
    {
        "exemplarCity";
        "וואגאדוגו";
    }
    "Porto-Novo";
    {
        "exemplarCity";
        "פורטו נובו";
    }
    "Sao_Tome";
    {
        "exemplarCity";
        "סאו טומה";
    }
    "Tripoli";
    {
        "exemplarCity";
        "טריפולי";
    }
    "Tunis";
    {
        "exemplarCity";
        "תוניס";
    }
    "Windhoek";
    {
        "exemplarCity";
        "וינדהוק";
    }
    }
    "Asia";
    {
        "Aden";
        {

```

```
        "exemplarCity";
        "עדן";
    }
    "Almaty";
    {
        "exemplarCity";
        "אלמאטי";
    }
    "Amman";
    {
        "exemplarCity";
        "עמאן";
    }
    "Anadyr";
    {
        "exemplarCity";
        "אנדיר";
    }
    "Aqtai";
    {
        "exemplarCity";
        "אקטאי";
    }
    "Aqtobe";
    {
        "exemplarCity";
        "אקטובה";
    }
    "Ashgabat";
    {
        "exemplarCity";
        "אשגבט";
    }
    "Baghdad";
    {
        "exemplarCity";
        "בגדד";
    }
    "Bahrain";
    {
        "exemplarCity";
        "בחריין";
    }
    "Baku";
    {
        "exemplarCity";
        "באקו";
    }
    "Bangkok";
    {
        "exemplarCity";
        "בנגקוק";
    }
    "Barnaul";
    {
        "exemplarCity";
        "ברנאול";
    }
```

```
}
"Beirut";
{
  "exemplarCity";
  "בֵּירוּת";
}
"Bishkek";
{
  "exemplarCity";
  "בִּישְׁקֵק";
}
"Brunei";
{
  "exemplarCity";
  "בְּרוּנֵי";
}
"Calcutta";
{
  "exemplarCity";
  "קוֹלְקָטָה";
}
"Chita";
{
  "exemplarCity";
  "צ'ִיטָה";
}
"Choibalsan";
{
  "exemplarCity";
  "צ'וֹיבַלְסָן";
}
"Colombo";
{
  "exemplarCity";
  "קוֹלוֹמְבוֹ";
}
"Damascus";
{
  "exemplarCity";
  "דַּמַּשֵּׁק";
}
"Dhaka";
{
  "exemplarCity";
  "דַּאֲקָה";
}
"Dili";
{
  "exemplarCity";
  "דִּילִי";
}
"Dubai";
{
  "exemplarCity";
  "דּוּבַאִי";
}
"Dushanbe";
```

```
{
  "exemplarCity";
  "דושנבה";
}
"Gaza";
{
  "exemplarCity";
  "עזה";
}
"Hebron";
{
  "exemplarCity";
  "חברון";
}
"Hong_Kong";
{
  "exemplarCity";
  "הונג קונג";
}
"Hovd";
{
  "exemplarCity";
  "חובד";
}
"Irkutsk";
{
  "exemplarCity";
  "אירקוטסק";
}
"Jakarta";
{
  "exemplarCity";
  "ג'קרטה";
}
"Jayapura";
{
  "exemplarCity";
  "ג'איאפורה";
}
"Jerusalem";
{
  "exemplarCity";
  "ירושלים";
}
"Kabul";
{
  "exemplarCity";
  "קאבול";
}
"Kamchatka";
{
  "exemplarCity";
  "קמצ'טקה";
}
"Karachi";
{
  "exemplarCity";
```



```

        "קראצ'י";
    }
    "Katmandu";
    {
        "exemplarCity";
        "קטמנדו";
    }
    "Khandyga";
    {
        "exemplarCity";
        "חנדיגה";
    }
    "Krasnoyarsk";
    {
        "exemplarCity";
        "קרטנויארסק";
    }
    "Kuala_Lumpur";
    {
        "exemplarCity";
        "קואלה לומפור";
    }
    "Kuching";
    {
        "exemplarCity";
        "קוצ'ינג";
    }
    "Kuwait";
    {
        "exemplarCity";
        "כווית";
    }
    "Macau";
    {
        "exemplarCity";
        "מקאו";
    }
    "Magadan";
    {
        "exemplarCity";
        "מגדל";
    }
    "Makassar";
    {
        "exemplarCity";
        "מאקאסאר";
    }
    "Manila";
    {
        "exemplarCity";
        "מנילה";
    }
    "Muscat";
    {
        "exemplarCity";
        "מוסקט";
    }
}

```

```
"Nicosia";
{
  "exemplarCity";
  "ניקוסיה";
}
"Novokuznetsk";
{
  "exemplarCity";
  "נובוקוזנטסק";
}
"Novosibirsk";
{
  "exemplarCity";
  "נובוסירסק";
}
"Omsk";
{
  "exemplarCity";
  "אומסק";
}
"Oral";
{
  "exemplarCity";
  "אורל";
}
"Phnom_Penh";
{
  "exemplarCity";
  "פנום פן";
}
"Pontianak";
{
  "exemplarCity";
  "פונטיאנק";
}
"Pyongyang";
{
  "exemplarCity";
  "פיונגיאנג";
}
"Qatar";
{
  "exemplarCity";
  "קטאר";
}
"Qyzylorda";
{
  "exemplarCity";
  "קזילורדה";
}
"Rangoon";
{
  "exemplarCity";
  "רנגון";
}
"Riyadh";
{
```

```
        "exemplarCity";
        "ריאד";
    }
    "Saigon";
    {
        "exemplarCity";
        "הו צ'י מין סיטי";
    }
    "Sakhalin";
    {
        "exemplarCity";
        "סחלין";
    }
    "Samarkand";
    {
        "exemplarCity";
        "סמרקנד";
    }
    "Seoul";
    {
        "exemplarCity";
        "סיאול";
    }
    "Shanghai";
    {
        "exemplarCity";
        "שנחאי";
    }
    "Singapore";
    {
        "exemplarCity";
        "סינגפור";
    }
    "Srednekolymsk";
    {
        "exemplarCity";
        "סרדנייקולימסק";
    }
    "Taipei";
    {
        "exemplarCity";
        "טאייפיי";
    }
    "Tashkent";
    {
        "exemplarCity";
        "טשקנט";
    }
    "Tbilisi";
    {
        "exemplarCity";
        "טביליסי";
    }
    "Tehran";
    {
        "exemplarCity";
        "טהרן";
    }
}
```

```
}
"Thimphu";
{
  "exemplarCity";
  "טהימפּהוּ";
}
"Tokyo";
{
  "exemplarCity";
  "טוקיו";
}
"Tomsk";
{
  "exemplarCity";
  "טומסק";
}
"Ulaanbaatar";
{
  "exemplarCity";
  "אולאאנבטאר";
}
"Urumqi";
{
  "exemplarCity";
  "אורומקי";
}
"Ust-Nera";
{
  "exemplarCity";
  "אוסט-נרה";
}
"Vientiane";
{
  "exemplarCity";
  "האנוי";
}
"Vladivostok";
{
  "exemplarCity";
  "ולדיװסטוק";
}
"Yakutsk";
{
  "exemplarCity";
  "יקוטסק";
}
"Yekaterinburg";
{
  "exemplarCity";
  "יקטרינבורג";
}
"Yerevan";
{
  "exemplarCity";
  "ירואן";
}
}
```

```
"Indian";
{
  "Antananarivo";
  {
    "exemplarCity";
    "אנטננריבו";
  }
  "Chagos";
  {
    "exemplarCity";
    "צ'אגוס";
  }
  "Christmas";
  {
    "exemplarCity";
    "האי כריסטמס";
  }
  "Cocos";
  {
    "exemplarCity";
    "קוקוס";
  }
  "Comoro";
  {
    "exemplarCity";
    "קומורו";
  }
  "Kerguelen";
  {
    "exemplarCity";
    "קרגוולן";
  }
  "Mahe";
  {
    "exemplarCity";
    "מהא";
  }
  "Maldives";
  {
    "exemplarCity";
    "האיים המלדיביים";
  }
  "Mauritius";
  {
    "exemplarCity";
    "מאוריציוס";
  }
  "Mayotte";
  {
    "exemplarCity";
    "מאיוט";
  }
  "Reunion";
  {
    "exemplarCity";
    "ראוניון";
  }
}
```

```
}
"Australia";
{
  "Adelaide";
  {
    "exemplarCity";
    "אדלייד";
  }
  "Brisbane";
  {
    "exemplarCity";
    "בריסביין";
  }
  "Broken_Hill";
  {
    "exemplarCity";
    "ברוקן היל";
  }
  "Currie";
  {
    "exemplarCity";
    "קרי";
  }
  "Darwin";
  {
    "exemplarCity";
    "דרווין";
  }
  "Eucla";
  {
    "exemplarCity";
    "יוקלה";
  }
  "Hobart";
  {
    "exemplarCity";
    "הוברט";
  }
  "Lindeman";
  {
    "exemplarCity";
    "לינדמן";
  }
  "Lord_Howe";
  {
    "exemplarCity";
    "אי הלורד האו";
  }
  "Melbourne";
  {
    "exemplarCity";
    "מלבורן";
  }
  "Perth";
  {
    "exemplarCity";
    "פרת';
  }
}
```

```
}
  "Sydney";
  {
    "exemplarCity";
    "סידני";
  }
}
"Pacific";
{
  "Apia";
  {
    "exemplarCity";
    "אפיה";
  }
  "Auckland";
  {
    "exemplarCity";
    "אוקלנד";
  }
  "Bougainville";
  {
    "exemplarCity";
    "בוגנוויל";
  }
  "Chatham";
  {
    "exemplarCity";
    "צ'אטהאם";
  }
  "Easter";
  {
    "exemplarCity";
    "אי הפסחא";
  }
  "Efate";
  {
    "exemplarCity";
    "אפטטה";
  }
  "Enderbury";
  {
    "exemplarCity";
    "אנדרבורי";
  }
  "Fakaofu";
  {
    "exemplarCity";
    "פקאופו";
  }
  "Fiji";
  {
    "exemplarCity";
    "פיג'י";
  }
  "Funafuti";
  {
    "exemplarCity";
```

```
        "פונפוטטי";
    }
    "Galapagos";
    {
        "exemplarCity";
        "גלפאגוס";
    }
    "Gambier";
    {
        "exemplarCity";
        "איי גמבייה";
    }
    "Guadalcanal";
    {
        "exemplarCity";
        "גוודלקנאל";
    }
    "Guam";
    {
        "exemplarCity";
        "גואם";
    }
    "Honolulu";
    {
        "exemplarCity";
        "הונולוּלוּ";
    }
    "Johnston";
    {
        "exemplarCity";
        "ג'ונסטון";
    }
    "Kiritimati";
    {
        "exemplarCity";
        "קיריטימאטי";
    }
    "Kosrae";
    {
        "exemplarCity";
        "קוסרה";
    }
    "Kwajalein";
    {
        "exemplarCity";
        "קוואג'ליין";
    }
    "Majuro";
    {
        "exemplarCity";
        "מאג'ורו";
    }
    "Marquesas";
    {
        "exemplarCity";
        "איי מרקייז";
    }
}
```



```
"Midway";
{
  "exemplarCity";
  "מידוויי";
}
"Nauru";
{
  "exemplarCity";
  "נאורו";
}
"Niue";
{
  "exemplarCity";
  "ניואה";
}
"Norfolk";
{
  "exemplarCity";
  "נורפוק";
}
"Noumea";
{
  "exemplarCity";
  "נומאה";
}
"Pago_Pago";
{
  "exemplarCity";
  "פאגו פאגו";
}
"Palau";
{
  "exemplarCity";
  "פלאו";
}
"Pitcairn";
{
  "exemplarCity";
  "פיטקרן";
}
"Ponape";
{
  "exemplarCity";
  "פונפיי";
}
"Port_Moresby";
{
  "exemplarCity";
  "פורט מורסבי";
}
"Rarotonga";
{
  "exemplarCity";
  "רארוטונגה";
}
"Saipan";
{
```

```
        "exemplarCity";
        "טאָיפּאַן";
    }
    "Tahiti";
    {
        "exemplarCity";
        "טהיטי";
    }
    "Tarawa";
    {
        "exemplarCity";
        "טאַראַוואַ";
    }
    "Tongatapu";
    {
        "exemplarCity";
        "טונגאַטאַפּו";
    }
    "Truk";
    {
        "exemplarCity";
        "צ'וק";
    }
    "Wake";
    {
        "exemplarCity";
        "וויק";
    }
    "Wallis";
    {
        "exemplarCity";
        "וואליס";
    }
}
"Arctic";
{
    "Longyearbyen";
    {
        "exemplarCity";
        "לונגייירבײן";
    }
}
"Antarctica";
{
    "Casey";
    {
        "exemplarCity";
        "קאַסיי";
    }
    "Davis";
    {
        "exemplarCity";
        "דיוויס";
    }
    "DumontDUrville";
    {
        "exemplarCity";
```

```

        "דומון ד'אורוויל";
    }
    "Macquarie";
    {
        "exemplarCity";
        "מקרי";
    }
    "Mawson";
    {
        "exemplarCity";
        "מוסון";
    }
    "McMurdo";
    {
        "exemplarCity";
        "מק-מרדו";
    }
    "Palmer";
    {
        "exemplarCity";
        "פאלמר";
    }
    "Rothera";
    {
        "exemplarCity";
        "רות'רה";
    }
    "Syowa";
    {
        "exemplarCity";
        "סיוואה";
    }
    "Troll";
    {
        "exemplarCity";
        "טרול";
    }
    "Vostok";
    {
        "exemplarCity";
        "ווסטוק";
    }
}
"Etc";
{
    "GMT";
    {
        "exemplarCity";
        "GMT";
    }
    "GMT1";
    {
        "exemplarCity";
        "GMT+1";
    }
    "GMT10";
    {

```

```
        "exemplarCity";
        "GMT+10";
    }
    "GMT11";
    {
        "exemplarCity";
        "GMT+11";
    }
    "GMT12";
    {
        "exemplarCity";
        "GMT+12";
    }
    "GMT2";
    {
        "exemplarCity";
        "GMT+2";
    }
    "GMT3";
    {
        "exemplarCity";
        "GMT+3";
    }
    "GMT4";
    {
        "exemplarCity";
        "GMT+4";
    }
    "GMT5";
    {
        "exemplarCity";
        "GMT+5";
    }
    "GMT6";
    {
        "exemplarCity";
        "GMT+6";
    }
    "GMT7";
    {
        "exemplarCity";
        "GMT+7";
    }
    "GMT8";
    {
        "exemplarCity";
        "GMT+8";
    }
    "GMT9";
    {
        "exemplarCity";
        "GMT+9";
    }
    "GMT-1";
    {
        "exemplarCity";
        "GMT-1";
    }
}
```

```
}
"GMT-10";
{
  "exemplarCity";
  "GMT-10";
}
"GMT-11";
{
  "exemplarCity";
  "GMT-11";
}
"GMT-12";
{
  "exemplarCity";
  "GMT-12";
}
"GMT-13";
{
  "exemplarCity";
  "GMT-13";
}
"GMT-14";
{
  "exemplarCity";
  "GMT-14";
}
"GMT-2";
{
  "exemplarCity";
  "GMT-2";
}
"GMT-3";
{
  "exemplarCity";
  "GMT-3";
}
"GMT-4";
{
  "exemplarCity";
  "GMT-4";
}
"GMT-5";
{
  "exemplarCity";
  "GMT-5";
}
"GMT-6";
{
  "exemplarCity";
  "GMT-6";
}
"GMT-7";
{
  "exemplarCity";
  "GMT-7";
}
"GMT-8";
```

```

        {
            "exemplarCity";
            "GMT-8";
        }
        "GMT-9";
        {
            "exemplarCity";
            "GMT-9";
        }
        "UTC";
        {
            "long";
            {
                "standard";
                "זמן אוניברסלי מתואם";
            }
            "short";
            {
                "standard";
                "UTC";
            }
            "exemplarCity";
            "UTC";
        }
        "Unknown";
        {
            "exemplarCity";
            "עיר לא ידועה";
        }
    }
    "metazone";
    {
        "Afghanistan";
        {
            "long";
            {
                "standard";
                "שעון אפגניסטן";
            }
        }
        "Africa_Central";
        {
            "long";
            {
                "standard";
                "שעון מרכז אפריקה";
            }
        }
        "Africa_Eastern";
        {
            "long";
            {
                "standard";
                "שעון מזרח אפריקה";
            }
        }
    }
}

```

```
"Africa_Southern";
{
  "long";
  {
    "standard";
    "שעון דרום אפריקה";
  }
}
"Africa_Western";
{
  "long";
  {
    "generic";
    "שעון מערב אפריקה",
    "standard";
    "שעון מערב אפריקה (חורף)",
    "daylight";
    "שעון מערב אפריקה (קיץ)";
  }
}
"Alaska";
{
  "long";
  {
    "generic";
    "שעון אלסקה",
    "standard";
    "שעון אלסקה (חורף)",
    "daylight";
    "שעון אלסקה (קיץ)";
  }
}
"Amazon";
{
  "long";
  {
    "generic";
    "שעון אמזונס",
    "standard";
    "שעון אמזונס (חורף)",
    "daylight";
    "שעון אמזונס (קיץ)";
  }
}
"America_Central";
{
  "long";
  {
    "generic";
    "שעון מרכז ארה"ב",
    "standard";
    "שעון מרכז ארה"ב (חורף)",
    "daylight";
    "שעון מרכז ארה"ב (קיץ)";
  }
}
"America_Eastern";
```

```
{
  "long";
  {
    "generic";
    "שעון החוף המזרחי",
    "standard";
    "שעון החוף המזרחי (חורף)",
    "daylight";
    "שעון החוף המזרחי (קיץ)";
  }
}
"America_Mountain";
{
  "long";
  {
    "generic";
    "שעון אזור ההרים בארה"ב",
    "standard";
    "שעון אזור ההרים בארה"ב (חורף)",
    "daylight";
    "שעון אזור ההרים בארה"ב (קיץ)";
  }
}
"America_Pacific";
{
  "long";
  {
    "generic";
    "שעון מערב ארה"ב",
    "standard";
    "שעון מערב ארה"ב (חורף)",
    "daylight";
    "שעון מערב ארה"ב (קיץ)";
  }
}
"Anadyr";
{
  "long";
  {
    "generic";
    "שעון אנדיר",
    "standard";
    "שעון רגיל אנדיר",
    "daylight";
    "שעון קיץ אנדיר";
  }
}
"Apia";
{
  "long";
  {
    "generic";
    "שעון אפיה",
    "standard";
    "שעון אפיה (חורף)",
    "daylight";
    "שעון אפיה (קיץ)";
  }
}
```



```
    }  
  }  
  "Arabian";  
  {  
    "long";  
    {  
      "generic";  
      "שעון חצי האי ערב",  
      "standard";  
      "שעון חצי האי ערב (חורף)",  
      "daylight";  
      "שעון חצי האי ערב (קיץ)";  
    }  
  }  
  "Argentina";  
  {  
    "long";  
    {  
      "generic";  
      "שעון ארגנטינה",  
      "standard";  
      "שעון ארגנטינה (חורף)",  
      "daylight";  
      "שעון ארגנטינה (קיץ)";  
    }  
  }  
  "Argentina_Western";  
  {  
    "long";  
    {  
      "generic";  
      "שעון מערב ארגנטינה",  
      "standard";  
      "שעון מערב ארגנטינה (חורף)",  
      "daylight";  
      "שעון מערב ארגנטינה (קיץ)";  
    }  
  }  
  "Armenia";  
  {  
    "long";  
    {  
      "generic";  
      "שעון ארמניה",  
      "standard";  
      "שעון ארמניה (חורף)",  
      "daylight";  
      "שעון ארמניה (קיץ)";  
    }  
  }  
  "Atlantic";  
  {  
    "long";  
    {  
      "generic";  
      "שעון האוקיינוס האטלנטי",  
      "standard";  
    }  
  }
```

```

        "שעון האוקיינוס האטלנטי (חורף)",
        "daylight";
        "שעון האוקיינוס האטלנטי (קיץ)";
    }
}
"Australia_Central";
{
    "long";
    {
        "generic";
        "שעון מרכז אוסטרליה",
        "standard";
        "שעון מרכז אוסטרליה (חורף)",
        "daylight";
        "שעון מרכז אוסטרליה (קיץ)";
    }
}
"Australia_CentralWestern";
{
    "long";
    {
        "generic";
        "שעון מרכז-מערב אוסטרליה",
        "standard";
        "שעון מרכז-מערב אוסטרליה (חורף)",
        "daylight";
        "שעון מרכז-מערב אוסטרליה (קיץ)";
    }
}
"Australia_Eastern";
{
    "long";
    {
        "generic";
        "שעון מזרח אוסטרליה",
        "standard";
        "שעון מזרח אוסטרליה (חורף)",
        "daylight";
        "שעון מזרח אוסטרליה (קיץ)";
    }
}
"Australia_Western";
{
    "long";
    {
        "generic";
        "שעון מערב אוסטרליה",
        "standard";
        "שעון מערב אוסטרליה (חורף)",
        "daylight";
        "שעון מערב אוסטרליה (קיץ)";
    }
}
"Azerbaijan";
{
    "long";
    {

```

```
        "generic";
        "שעון אזרבייג'אן",
        "standard";
        "שעון אזרבייג'אן (חורף)",
        "daylight";
        "שעון אזרבייג'אן (קיץ)";
    }
}
"Azores";
{
    "long";
    {
        "generic";
        "שעון האיים האזוריים",
        "standard";
        "שעון האיים האזוריים (חורף)",
        "daylight";
        "שעון האיים האזוריים (קיץ)";
    }
}
"Bangladesh";
{
    "long";
    {
        "generic";
        "שעון בנגלדש",
        "standard";
        "שעון בנגלדש (חורף)",
        "daylight";
        "שעון בנגלדש (קיץ)";
    }
}
"Bhutan";
{
    "long";
    {
        "standard";
        "שעון בהוטן";
    }
}
"Bolivia";
{
    "long";
    {
        "standard";
        "שעון בוליביה";
    }
}
"Brasilia";
{
    "long";
    {
        "generic";
        "שעון ברזיליה",
        "standard";
        "שעון ברזיליה (חורף)",
        "daylight";
    }
}
```

```

        "שעון ברזיליה (קיץ)";
    }
}
"Brunei";
{
    "long";
    {
        "standard";
        "שעון ברוניי דארוסלאם";
    }
}
"Cape_Verde";
{
    "long";
    {
        "generic";
        "שעון כף ורדה",
        "standard";
        "שעון כף ורדה (חורף)",
        "daylight";
        "שעון כף ורדה (קיץ)";
    }
}
"Chamorro";
{
    "long";
    {
        "standard";
        "שעון צ'אמורו";
    }
}
"Chatham";
{
    "long";
    {
        "generic";
        "שעון צ'טהאם",
        "standard";
        "שעון צ'טהאם (חורף)",
        "daylight";
        "שעון צ'טהאם (קיץ)";
    }
}
"Chile";
{
    "long";
    {
        "generic";
        "שעון צ'ילה",
        "standard";
        "שעון צ'ילה (חורף)",
        "daylight";
        "שעון צ'ילה (קיץ)";
    }
}
"China";
{

```

```
        "long";
        {
            "generic";
            "שעון סין",
            "standard";
            "שעון סין (חורף)",
            "daylight";
            "שעון סין (קיץ)";
        }
    }
    "Choibalsan";
    {
        "long";
        {
            "generic";
            "שעון צ'ויבלסן",
            "standard";
            "שעון צ'ויבלסן (חורף)",
            "daylight";
            "שעון צ'ויבלסן (קיץ)";
        }
    }
    "Christmas";
    {
        "long";
        {
            "standard";
            "שעון האי כריסטמס";
        }
    }
    "Cocos";
    {
        "long";
        {
            "standard";
            "שעון איי קוקוס";
        }
    }
    "Colombia";
    {
        "long";
        {
            "generic";
            "שעון קולומביה",
            "standard";
            "שעון קולומביה (חורף)",
            "daylight";
            "שעון קולומביה (קיץ)";
        }
    }
    "Cook";
    {
        "long";
        {
            "generic";
            "שעון איי קוק",
            "standard";
        }
    }
}
```

```

        "שעון איי קוק (חורף)",
        "daylight";
        "שעון איי קוק (מחצית הקיץ)";
    }
}
"Cuba";
{
    "long";
    {
        "generic";
        "שעון קובה",
        "standard";
        "שעון קובה (חורף)",
        "daylight";
        "שעון קובה (קיץ)";
    }
}
"Davis";
{
    "long";
    {
        "standard";
        "שעון דייוויס";
    }
}
"DumontDUrville";
{
    "long";
    {
        "standard";
        "שעון דומון ד'אורוויל";
    }
}
"East_Timor";
{
    "long";
    {
        "standard";
        "שעון מזרח טימור";
    }
}
"Easter";
{
    "long";
    {
        "generic";
        "שעון אי הפסחא",
        "standard";
        "שעון אי הפסחא (חורף)",
        "daylight";
        "שעון אי הפסחא (קיץ)";
    }
}
"Ecuador";
{
    "long";
    {

```

```

        "standard";
        "שעון אקוודור";
    }
}
"Europe_Central";
{
    "long";
    {
        "generic";
        "שעון מרכז אירופה",
        "standard";
        "שעון מרכז אירופה (חורף)",
        "daylight";
        "שעון מרכז אירופה (קיץ)";
    }
}
"Europe_Eastern";
{
    "long";
    {
        "generic";
        "שעון מזרח אירופה",
        "standard";
        "שעון מזרח אירופה (חורף)",
        "daylight";
        "שעון מזרח אירופה (קיץ)";
    }
}
"Europe_Further_Eastern";
{
    "long";
    {
        "standard";
        "שעון מינסק";
    }
}
"Europe_Western";
{
    "long";
    {
        "generic";
        "שעון מערב אירופה",
        "standard";
        "שעון מערב אירופה (חורף)",
        "daylight";
        "שעון מערב אירופה (קיץ)";
    }
}
"Falkland";
{
    "long";
    {
        "generic";
        "שעון איי פוקלנד",
        "standard";
        "שעון איי פוקלנד (חורף)",
        "daylight";
    }
}

```

```
        "שעון איי פוקלנד (קיץ)";
    }
}
"Fiji";
{
    "long";
    {
        "generic";
        "שעון פיג'י",
        "standard";
        "שעון פיג'י (חורף)",
        "daylight";
        "שעון פיג'י (קיץ)";
    }
}
"French_Guiana";
{
    "long";
    {
        "standard";
        "שעון גיאנה הצרפתית";
    }
}
"French_Southern";
{
    "long";
    {
        "standard";
        "שעון הארצות הדרומיות והאנטארקטיות של צרפת";
    }
}
"Galapagos";
{
    "long";
    {
        "standard";
        "שעון איי גלאפגוס";
    }
}
"Gambier";
{
    "long";
    {
        "standard";
        "שעון איי גמביה";
    }
}
"Georgia";
{
    "long";
    {
        "generic";
        "שעון גאורגיה",
        "standard";
        "שעון גאורגיה (חורף)",
        "daylight";
        "שעון גאורגיה (קיץ)";
    }
}
```



```

    }
  }
  "Gilbert_Islands";
  {
    "long";
    {
      "standard";
      "שעון איי גילברט";
    }
  }
  "GMT";
  {
    "long";
    {
      "standard";
      "שעון גריניץ";
    }
  }
  "Greenland_Eastern";
  {
    "long";
    {
      "generic";
      "שעון מזרח גרינלנד",
      "standard";
      "שעון מזרח גרינלנד (חורף)",
      "daylight";
      "שעון מזרח גרינלנד (קיץ)";
    }
  }
  "Greenland_Western";
  {
    "long";
    {
      "generic";
      "שעון מערב גרינלנד",
      "standard";
      "שעון מערב גרינלנד (חורף)",
      "daylight";
      "שעון מערב גרינלנד (קיץ)";
    }
  }
  "Gulf";
  {
    "long";
    {
      "standard";
      "שעון מדינות המפרץ";
    }
  }
  "Guyana";
  {
    "long";
    {
      "standard";
      "שעון גיאנה";
    }
  }

```

```
}
"Hawaii_Aleutian";
{
  "long";
  {
    "generic";
    "שעון האיים האלאוטיים הוואי",
    "standard";
    "שעון האיים האלאוטיים הוואי (חורף)",
    "daylight";
    "שעון האיים האלאוטיים הוואי (קיץ)";
  }
}
"Hong_Kong";
{
  "long";
  {
    "generic";
    "שעון הונג קונג",
    "standard";
    "שעון הונג קונג (חורף)",
    "daylight";
    "שעון הונג קונג (קיץ)";
  }
}
"Hovd";
{
  "long";
  {
    "generic";
    "שעון חובד",
    "standard";
    "שעון חובד (חורף)",
    "daylight";
    "שעון חובד (קיץ)";
  }
}
"India";
{
  "long";
  {
    "standard";
    "שעון הודו";
  }
}
"Indian_Ocean";
{
  "long";
  {
    "standard";
    "שעון האוקיינוס ההודי";
  }
}
"Indochina";
{
  "long";
  {
```

```

        "standard";
        "שעון הודו-סין";
    }
}
"Indonesia_Central";
{
    "long";
    {
        "standard";
        "שעון מרכז אינדונזיה";
    }
}
"Indonesia_Eastern";
{
    "long";
    {
        "standard";
        "שעון מזרח אינדונזיה";
    }
}
"Indonesia_Western";
{
    "long";
    {
        "standard";
        "שעון מערב אינדונזיה";
    }
}
"Iran";
{
    "long";
    {
        "generic";
        "שעון איראן",
        "standard";
        "שעון איראן (חורף)",
        "daylight";
        "שעון איראן (קיץ)";
    }
}
"Irkutsk";
{
    "long";
    {
        "generic";
        "שעון אירקוטסק",
        "standard";
        "שעון אירקוטסק (חורף)",
        "daylight";
        "שעון אירקוטסק (קיץ)";
    }
}
"Israel";
{
    "long";
    {
        "generic";

```

```

        "שעון ישראל",
        "standard";
        "שעון ישראל (חורף)",
        "daylight";
        "שעון ישראל (קיץ)";
    }
}
"Japan";
{
    "long";
    {
        "generic";
        "שעון יפן",
        "standard";
        "שעון יפן (חורף)",
        "daylight";
        "שעון יפן (קיץ)";
    }
}
"Kamchatka";
{
    "long";
    {
        "generic";
        "שעון פטרופב-קמצ'טסק",
        "standard";
        "שעון רגיל פטרופב-קמצ'טסק",
        "daylight";
        "שעון קיץ פטרופב-קמצ'טסק";
    }
}
"Kazakhstan_Eastern";
{
    "long";
    {
        "standard";
        "שעון מזרח קזחסטן";
    }
}
"Kazakhstan_Western";
{
    "long";
    {
        "standard";
        "שעון מערב קזחסטן";
    }
}
"Korea";
{
    "long";
    {
        "generic";
        "שעון קוריאה",
        "standard";
        "שעון קוריאה (חורף)",
        "daylight";
        "שעון קוריאה (קיץ)";
    }
}

```

```
    }  
  }  
  "Kosrae";  
  {  
    "long";  
    {  
      "standard";  
      "שעון קוסראה";  
    }  
  }  
  "Krasnoyarsk";  
  {  
    "long";  
    {  
      "generic";  
      "שעון קרסנויארסק",  
      "standard";  
      "שעון קרסנויארסק (חורף)",  
      "daylight";  
      "שעון קרסנויארסק (קיץ)";  
    }  
  }  
  "Kyrgystan";  
  {  
    "long";  
    {  
      "standard";  
      "שעון קירגיזסטן";  
    }  
  }  
  "Line_Islands";  
  {  
    "long";  
    {  
      "standard";  
      "שעון איי ליין";  
    }  
  }  
  "Lord_Howe";  
  {  
    "long";  
    {  
      "generic";  
      "שעון אי הלורד האו",  
      "standard";  
      "שעון אי הלורד האו (חורף)",  
      "daylight";  
      "שעון אי הלורד האו (קיץ)";  
    }  
  }  
  "Macau";  
  {  
    "long";  
    {  
      "generic";  
      "שעון מקאו",  
      "standard";  
    }  
  }
```

```
        "שעון חורף מקאו",
        "daylight";
        "שעון קיץ מקאו";
    }
}
"Macquarie";
{
    "long";
    {
        "standard";
        "שעון מקווארי";
    }
}
"Magadan";
{
    "long";
    {
        "generic";
        "שעון מגדן",
        "standard";
        "שעון מגדן (חורף)",
        "daylight";
        "שעון מגדן (קיץ)";
    }
}
"Malaysia";
{
    "long";
    {
        "standard";
        "שעון מלזיה";
    }
}
"Maldives";
{
    "long";
    {
        "standard";
        "שעון האיים המלדיביים";
    }
}
"Marquesas";
{
    "long";
    {
        "standard";
        "שעון איי מרקז";
    }
}
"Marshall_Islands";
{
    "long";
    {
        "standard";
        "שעון איי מרשל";
    }
}
}
```

```
"Mauritius";
{
  "long";
  {
    "generic";
    "שעון מאוריצייוס",
    "standard";
    "שעון מאוריצייוס (חורף)",
    "daylight";
    "שעון מאוריצייוס (קיץ)";
  }
}
"Mawson";
{
  "long";
  {
    "standard";
    "שעון מאוסון";
  }
}
"Mexico_Northwest";
{
  "long";
  {
    "generic";
    "שעון צפון-מערב מקסיקו",
    "standard";
    "שעון צפון-מערב מקסיקו (חורף)",
    "daylight";
    "שעון צפון-מערב מקסיקו (קיץ)";
  }
}
"Mexico_Pacific";
{
  "long";
  {
    "generic";
    "שעון מערב מקסיקו",
    "standard";
    "שעון מערב מקסיקו (חורף)",
    "daylight";
    "שעון מערב מקסיקו (קיץ)";
  }
}
"Mongolia";
{
  "long";
  {
    "generic";
    "שעון אולן בטור",
    "standard";
    "שעון אולן בטור (חורף)",
    "daylight";
    "שעון אולן בטור (קיץ)";
  }
}
"Moscow";
```

```
{
  "long";
  {
    "generic";
    "שעון מוסקבה",
    "standard";
    "שעון מוסקבה (חורף)",
    "daylight";
    "שעון מוסקבה (קיץ)";
  }
}
"Myanmar";
{
  "long";
  {
    "standard";
    "שעון מיאנמר";
  }
}
"Nauru";
{
  "long";
  {
    "standard";
    "שעון נאורו";
  }
}
"Nepal";
{
  "long";
  {
    "standard";
    "שעון נפאל";
  }
}
"New_Caledonia";
{
  "long";
  {
    "generic";
    "שעון קלדוניה החדשה",
    "standard";
    "שעון קלדוניה החדשה (חורף)",
    "daylight";
    "שעון קלדוניה החדשה (קיץ)";
  }
}
"New_Zealand";
{
  "long";
  {
    "generic";
    "שעון ניו זילנד",
    "standard";
    "שעון ניו זילנד (חורף)",
    "daylight";
    "שעון ניו זילנד (קיץ)";
  }
}
```



```
    }  
  }  
  "Newfoundland";  
  {  
    "long";  
    {  
      "generic";  
      "שעון ניופאונדלנד",  
      "standard";  
      "שעון ניופאונדלנד (חורף)",  
      "daylight";  
      "שעון ניופאונדלנד (קיץ)";  
    }  
  }  
  "Niue";  
  {  
    "long";  
    {  
      "standard";  
      "שעון ניואה";  
    }  
  }  
  "Norfolk";  
  {  
    "long";  
    {  
      "standard";  
      "שעון האי נורפוק";  
    }  
  }  
  "Noronha";  
  {  
    "long";  
    {  
      "generic";  
      "שעון פרננדו די נורוניה",  
      "standard";  
      "שעון פרננדו די נורוניה (חורף)",  
      "daylight";  
      "שעון פרננדו די נורוניה (קיץ)";  
    }  
  }  
  "Novosibirsk";  
  {  
    "long";  
    {  
      "generic";  
      "שעון נובוסיבירסק",  
      "standard";  
      "שעון נובוסיבירסק (חורף)",  
      "daylight";  
      "שעון נובוסיבירסק (קיץ)";  
    }  
  }  
  "Omsk";  
  {  
    "long";
```

```

        {
            "generic";
            "שעון אומסק",
            "standard";
            "שעון אומסק (חורף)",
            "daylight";
            "שעון אומסק (קיץ)";
        }
    }
    "Pakistan";
    {
        "long";
        {
            "generic";
            "שעון פקיסטן",
            "standard";
            "שעון פקיסטן (חורף)",
            "daylight";
            "שעון פקיסטן (קיץ)";
        }
    }
    "Palau";
    {
        "long";
        {
            "standard";
            "שעון פלאו";
        }
    }
    "Papua_New_Guinea";
    {
        "long";
        {
            "standard";
            "שעון פפואה גיניאה החדשה";
        }
    }
    "Paraguay";
    {
        "long";
        {
            "generic";
            "שעון פרגוואי",
            "standard";
            "שעון פרגוואי (חורף)",
            "daylight";
            "שעון פרגוואי (קיץ)";
        }
    }
    "Peru";
    {
        "long";
        {
            "generic";
            "שעון פרו",
            "standard";
            "שעון פרו (חורף)",

```

```

        "daylight";
        "שעון פרו (קיץ)";
    }
}
"Philippines";
{
    "long";
    {
        "generic";
        "שעון הפיליפינים",
        "standard";
        "שעון הפיליפינים (חורף)",
        "daylight";
        "שעון הפיליפינים (קיץ)";
    }
}
"Phoenix_Islands";
{
    "long";
    {
        "standard";
        "שעון איי פיניקס";
    }
}
"Pierre_Miquelon";
{
    "long";
    {
        "generic";
        "שעון סנט פייר ומיקלון",
        "standard";
        "שעון סנט פייר ומיקלון (חורף)",
        "daylight";
        "שעון סנט פייר ומיקלון (קיץ)";
    }
}
"Pitcairn";
{
    "long";
    {
        "standard";
        "שעון פיטקרן";
    }
}
"Ponape";
{
    "long";
    {
        "standard";
        "שעון פונאפי";
    }
}
"Pyongyang";
{
    "long";
    {
        "standard";

```

```
        "שעון פיונגיאנג";
    }
}
"Reunion";
{
    "long";
    {
        "standard";
        "שעון ראוניון";
    }
}
"Rothera";
{
    "long";
    {
        "standard";
        "שעון רות'רה";
    }
}
"Sakhalin";
{
    "long";
    {
        "generic";
        "שעון סחלין",
        "standard";
        "שעון סחלין (חורף)",
        "daylight";
        "שעון סחלין (קיץ)";
    }
}
"Samara";
{
    "long";
    {
        "generic";
        "שעון סמרה",
        "standard";
        "שעון רגיל סמרה",
        "daylight";
        "שעון קיץ סמרה";
    }
}
"Samoa";
{
    "long";
    {
        "generic";
        "שעון סמואה",
        "standard";
        "שעון סמואה (חורף)",
        "daylight";
        "שעון סמואה (קיץ)";
    }
}
"Seychelles";
{
```

```
        "long";
        {
            "standard";
            "שעון איי סיישל";
        }
    }
    "Singapore";
    {
        "long";
        {
            "standard";
            "שעון סינגפור";
        }
    }
    "Solomon";
    {
        "long";
        {
            "standard";
            "שעון איי שלמה";
        }
    }
    "South_Georgia";
    {
        "long";
        {
            "standard";
            "שעון דרום ג'ורג'יה";
        }
    }
    "Suriname";
    {
        "long";
        {
            "standard";
            "שעון סורינאם";
        }
    }
    "Syowa";
    {
        "long";
        {
            "standard";
            "שעון סייווה";
        }
    }
    "Tahiti";
    {
        "long";
        {
            "standard";
            "שעון טהיטי";
        }
    }
    "Taipei";
    {
        "long";
```

```

        {
            "generic";
            "שעון טאיפיי",
            "standard";
            "שעון טאיפיי (חורף)",
            "daylight";
            "שעון טאיפיי (קיץ)";
        }
    }
    "Tajikistan";
    {
        "long";
        {
            "standard";
            "שעון טג'יקיסטן";
        }
    }
    "Tokelau";
    {
        "long";
        {
            "standard";
            "שעון טוקלאו";
        }
    }
    "Tonga";
    {
        "long";
        {
            "generic";
            "שעון טונגה",
            "standard";
            "שעון טונגה (חורף)",
            "daylight";
            "שעון טונגה (קיץ)";
        }
    }
    "Truk";
    {
        "long";
        {
            "standard";
            "שעון צ'וק";
        }
    }
    "Turkmenistan";
    {
        "long";
        {
            "generic";
            "שעון טורקמניסטן",
            "standard";
            "שעון טורקמניסטן (חורף)",
            "daylight";
            "שעון טורקמניסטן (קיץ)";
        }
    }
}

```

```
"Tuvalu";
{
  "long";
  {
    "standard";
    "שעון טובאלו";
  }
}
"Uruguay";
{
  "long";
  {
    "generic";
    "שעון אורוגוואי",
    "standard";
    "שעון אורוגוואי (חורף)",
    "daylight";
    "שעון אורוגוואי (קיץ)";
  }
}
"Uzbekistan";
{
  "long";
  {
    "generic";
    "שעון אוזבקיסטן",
    "standard";
    "שעון אוזבקיסטן (חורף)",
    "daylight";
    "שעון אוזבקיסטן (קיץ)";
  }
}
"Vanuatu";
{
  "long";
  {
    "generic";
    "שעון ונואטו",
    "standard";
    "שעון ונואטו (חורף)",
    "daylight";
    "שעון ונואטו (קיץ)";
  }
}
"Venezuela";
{
  "long";
  {
    "standard";
    "שעון ונצואלה";
  }
}
"Vladivostok";
{
  "long";
  {
    "generic";
```

```

        "שעון ולדיווסטוק",
        "standard";
        "שעון ולדיווסטוק (חורף)",
        "daylight";
        "שעון ולדיווסטוק (קיץ)";
    }
}
"Volgograd";
{
    "long";
    {
        "generic";
        "שעון וולגוגרד",
        "standard";
        "שעון וולגוגרד (חורף)",
        "daylight";
        "שעון וולגוגרד (קיץ)";
    }
}
"Vostok";
{
    "long";
    {
        "standard";
        "שעון ווסטוק";
    }
}
"Wake";
{
    "long";
    {
        "standard";
        "שעון האי וייק";
    }
}
"Wallis";
{
    "long";
    {
        "standard";
        "שעון וואליס ופוטונה";
    }
}
"Yakutsk";
{
    "long";
    {
        "generic";
        "שעון יקוטסק",
        "standard";
        "שעון יקוטסק (חורף)",
        "daylight";
        "שעון יקוטסק (קיץ)";
    }
}
"Yekaterinburg";
{

```



```
        "long";  
    {  
        "generic";  
        "שעון יקטרינבורג",  
        "standard";  
        "שעון יקטרינבורג (חורף)",  
        "daylight";  
        "שעון יקטרינבורג (קיץ)";  
    }  
}  
  
}  
  
}  
  
}
```

Date Format

Date format is a way of representing the start date, end date strings in different string format in the textbox.

By default, the `DateTimeRangePicker`'s start and end date string format is based on the culture. You can also set the own

custom format by using the [format](#) property.

Once the date format property has been defined it will be common to all the cultures.

To know more about the date format standards, refer to the [Internationalization Date Format](#) section.

The following example demonstrates the `DateRangePicker` with the custom format `(yyyy-MM-dd)`. Also, here the separator of the date values is changed to string "to".

[Class-component]

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
//import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
class App extends React.Component {
  render() {
    return <DateRangePickerComponent id="daterangepicker" format='yyyy-MM-dd' separator='to' placeholder='Select a range'/>;
  }
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
//import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
```

```
class App extends React.Component<{}, {}> {
  render() {
    return <DateRangePickerComponent id="daterangepicker" format='yyyy-MM-dd' separator='to' placeholder='Select a range' />
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

[Functional-component]

INDEX.JSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
//import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
function App() {
  return <DateRangePickerComponent id="daterangepicker" format='yyyy-MM-dd' separator='to' placeholder='Select a range' />;
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
import * as React from "react";
import * as ReactDOM from "react-dom";
//import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
function App() {
  return <DateRangePickerComponent id="daterangepicker" format='yyyy-MM-dd' separator='to' placeholder='Select a range' />;
};
ReactDOM.render(<App />, document.getElementById('element'));
```

Customization in React Daterangepicker component

DateRangePicker makes available for the UI customization which can be achieved with events, properties that are available with this component.

Day cell format

[renderDayCell](#) is a event which provides the option to customize each day cell while rendering itself.

The following example disables the weekends of every month using [renderDayCell](#) event. Here we have used the `e-disabled` class to highlight the disabled date

[Class-component]

INDEX.JSX

```
// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
  onRenderDayCell(args) {
```

```

        if (args.date.getDay() === 0 || args.date.getDay() === 6) {
            // sets isDisabled to true to disable the date.
            args.isDisabled = true;
        }
    }
    render() {
        return <DateRangePickerComponent renderDayCell={this.onRenderDayCell}
placeholder='Select a range' />;
    }
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

// import the daterangepicker component
import { DateRangePickerComponent, RenderDayCellEventArgs } from
'@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
    public onRenderDayCell(args: RenderDayCellEventArgs): void {
        if ((args.date as Date).getDay() === 0 || (args.date as
Date).getDay() === 6) {
            // sets isDisabled to true to disable the date.
            args.isDisabled = true;
        }
    }
    public render() {
        return <DateRangePickerComponent renderDayCell={this.onRenderDayCell}
placeholder='Select a range' />
    }
};
ReactDOM.render(<App />, document.getElementById('element'));

```

[Functional-component]

INDEX.JSX

```

// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    function onRenderDayCell(args) {
        if (args.date.getDay() === 0 || args.date.getDay() === 6) {
            // sets isDisabled to true to disable the date.
            args.isDisabled = true;
        }
    }
    return <DateRangePickerComponent renderDayCell={onRenderDayCell}
placeholder='Select a range' />;
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```
// import the daterangepicker component
import { DateRangePickerComponent, RenderDayCellEventArgs } from
 '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    function onRenderDayCell(args: RenderDayCellEventArgs): void {
        if ((args.date as Date).getDay() === 0 || (args.date as
        Date).getDay() === 6) {
            // sets isDisabled to true to disable the date.
            args.isDisabled = true;
        }
    }
    return <DateRangePickerComponent renderDayCell={onRenderDayCell}
    placeholder='Select a range' />
};
ReactDOM.render(<App />, document.getElementById('element'));
```

First day of week

Start day in a week will differ based on culture. But, you can customize this based on application needs also. For this, you can make use of `firstDayOfWeek` property. By default, first day of week in en-US is a Sunday.

In following sample, it customized to Monday with help of this property.

[Class-component]**INDEX.JSX**

```
// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
    startDay = 1;
    render() {
        return <DateRangePickerComponent firstDayOfWeek={this.startDay}
        placeholder='Select a range' />;
    }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
    private startDay:number = 1;
    public render() {
        return <DateRangePickerComponent firstDayOfWeek={this.startDay}
        placeholder='Select a range' />
    }
};
```

```

    }
  };
  ReactDOM.render(<App />, document.getElementById('element'));

```

[Functional-component]

INDEX.JSX

```

// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  const startDay = 1;
  return <DateRangePickerComponent firstDayOfWeek={startDay}
placeholder='Select a range' />;
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  const startDay:number = 1;
  return <DateRangePickerComponent firstDayOfWeek={startDay}
placeholder='Select a range' />
};
ReactDOM.render(<App />, document.getElementById('element'));

```

Preset ranges

DateRangePicker provides an option to set the predefined ranges in a DateRangePicker via [presets](#) properties with the corresponding label. This property will accept the values in the order of label, start date (date object), end date (date object) and append these ranges in a component for quick selection.

In following sample, you can choose the frequently using ranges options from the list of ranges itself easily.

[Class-component]

INDEX.JSX

```

// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
  presets = [
    { label: 'Today', start: new Date(), end: new Date() },
    { label: 'Last Week', start: new Date(new Date().setDate(new
Date().getDate() - 7)), end: new Date() }
  ];
}

```

```

    render() {
        return <DateRangePickerComponent placeholder='Select a range'
presets={this.presets}/>;
    }
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

// import the daterangepicker component
import { DateRangePickerComponent, PresetsModel } from '@syncfusion/ej2-
react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
    private presets: PresetsModel[] = [
        { label: 'Today', start: new Date(), end: new Date() },
        { label: 'Last Week', start: new Date(new Date().setDate(new
Date().getDate() - 7)), end: new Date() }
    ];
    public render() {
        return <DateRangePickerComponent placeholder='Select a range'
presets={this.presets} />
    }
};
ReactDOM.render(<App />, document.getElementById('element'));

```

[Functional-component]**INDEX.JSX**

```

// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    const presets = [
        { label: 'Today', start: new Date(), end: new Date() },
        { label: 'Last Week', start: new Date(new Date().setDate(new
Date().getDate() - 7)), end: new Date() }
    ];
    return <DateRangePickerComponent placeholder='Select a range'
presets={presets}/>;
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

// import the daterangepicker component
import { DateRangePickerComponent, PresetsModel } from '@syncfusion/ej2-
react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";

```

```
function App() {
  const presets: PresetsModel[] = [
    { label: 'Today', start: new Date(), end: new Date() },
    { label: 'Last Week', start: new Date(new Date().setDate(new
Date().getDate() - 7)), end: new Date() }
  ];
  return <DateRangePickerComponent placeholder='Select a range '
presets={presets} />
};
ReactDOM.render(<App />, document.getElementById('element'));
```

See Also

- [How to customize DateRangePicker using cssClass](#)
- [How to disable DateRangePicker component](#)
- [How to customize the DateRangePicker day header](#)

Accessibility in React Daterangepicker component

The DateRangePicker component followed the accessibility guidelines and standards, including [ADA](#), [Section 508](#), [WCAG 2.2](#) standards, and [WCAG roles](#) that are commonly used to evaluate accessibility.

The accessibility compliance for the DateRangePicker component is outlined below.

| Accessibility Criteria | Compatibility |

| -- | -- |

| [WCAG 2.2 Support](#) | |

| [Section 508 Support](#) | |

| [Screen Reader Support](#) | |

| [Right-To-Left Support](#) | |

| [Color Contrast](#) | |

| [Mobile Device Support](#) | |

| [Keyboard Navigation Support](#) | |

| [Accessibility Checker Validation](#) | |

| [Axe-core Accessibility Validation](#) | |

<style>

```
.post .post-content img {
display: inline-block;
margin: 0.5em 0;
}
</style>

<div> - All
features of the component meet the requirement.</div>

<div> - Some features of the component do not meet the requirement.</div>

<div> - The component does not meet the requirement.</div>
```

WAI-ARIA attributes

The web accessibility makes web content and web applications more accessible for disabled people. It especially helps in dynamic content change and development of advanced user interface controls with AJAX, HTML, JavaScript, and related technologies.

DateRangePicker provides built-in compliance with [WAI-ARIA](#) specifications. WAI-ARIA supports is achieved through the attributes like `aria-expanded`, `aria-disabled`, `aria-activedescendant` applied to the input element.

To know about the accessibility of Calendar refer to the [Accessibility](#) section of Calendar.

It helps disabled persons by providing information about the widget for assistive technology in the screen readers.

DateRangePicker component contains grid role and grid cell for each day cell.

- **Aria-expanded:** Indicates the currently selected date of the DateRangePicker component.
- **Aria-disabled:** Indicates the disabled state of the DateRangePicker component.

Keyboard Interaction

You can use the following keys to interact with the DateRangePicker. This component implements the keyboard navigation support by following the [WAI-ARIA practices](#).

It supports the following list of shortcut keys:

Input Navigation

Before opening the popup, use the below list of keys to control the popup element.

| **Press** | **To do this** |

| --- | --- |

| **Alt + Down Arrow** | **Opens the popup.** |

| **Alt + Up Arrow** | **Closes the popup.** |

| **Esc** | **closes the popup (If the Component's input element is in focused state).** |

Calendar Navigation

Use the following list of keys to navigate the currently focused Calendar after the popup has opened.

| **Press** | **To do this** |

| --- | --- |

| **Upper Arrow** | **Focuses the same day of the previous week.** |

| **Down Arrow** | **Focuses the same day of the next week.** |

| **Left Arrow** | **Focuses the day before.** |

| **Right Arrow** | **Focuses the next day.** |

| **Home** | **Focuses the first day of the month.** |

| **End** | **Focuses the last day of the month.** |

| **Page Up** | **Focuses the same date of the previous month.** |

| **Page Down** | **Focuses the same date of the next month.** |

| **Enter** | **Selects the currently focused date.** |

| **Shift + Page Up** | **Focuses the same date for the previous year.** |

| **Shift + Page Down** | **Focuses the same date for the next year.** |

| **Control + Home** | **Focuses the first date of the current year.** |

| **Control + End** | **Focuses the last date of the current year.** |

| **Alt + Right** | **Focuses through out the pop-up container in forward direction.** |

| **Alt + Left** | **Focuses through out the pop-up container in backward direction.** |

To focus the DateRangePicker component, use the **alt+t** keys.

[Class-component]

INDEX.JSX

```
{% raw %}
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
  daterangeInstance;
  componentDidMount() {
    const proxy = this;
    document.onkeyup = (e) => {
      if (e.altKey && e.keyCode === 84 /* t */) {
        // press alt+t to focus the control.
        proxy.daterangeInstance.element.focus();
      }
    };
  }
  render() {
    return <DateRangePickerComponent placeholder="Select a range"
    ref={ (scope) => { this.daterangeInstance = scope; } } />;
  }
}
```

```

}
;
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}

```

INDEX.TSX

```

{% raw %}
import { DateRangePicker, DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
  private daterangeInstance: DateRangePicker;
  public componentDidMount() {
    const proxy = this;
    document.onkeyup = (e) => {
      if (e.altKey && e.keyCode === 84 /* t */) {
        // press alt+t to focus the control.
        proxy.daterangeInstance.element.focus();
      }
    };
  }
  public render() {
    return <DateRangePickerComponent placeholder="Select a range"
    ref={ (scope) => { (this.daterangeInstance as DateRangePicker | null) = scope;
  }} />
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}

```

[Functional-component]

INDEX.JSX

```

{% raw %}
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
  let daterangeInstance;
  React.useEffect(() => {
    const proxy = this;
    document.onkeyup = (e) => {
      if (e.altKey && e.keyCode === 84 /* t */) {
        // press alt+t to focus the control.
        proxy.daterangeInstance.element.focus();
      }
    };
  }, []);
  return <DateRangePickerComponent placeholder="Select a range"
  ref={ (scope) => { daterangeInstance = scope; }} />
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

```
{% endraw %}
```

INDEX.TSX

```
{% raw %}
import { DateRangePicker, DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
function App() {
    let daterangeInstance: DateRangePicker;
    React.useEffect(() => {
        const proxy = this;
        document.onkeyup = (e) => {
            if (e.altKey && e.keyCode === 84 /* t */) {
                // press alt+t to focus the control.
                proxy.daterangeInstance.element.focus();
            }
        };
    }, []);

    return <DateRangePickerComponent placeholder="Select a range"
ref={ (scope) => { (daterangeInstance as DateRangePicker | null) = scope; } } />
};
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}
```

Ensuring accessibility

The DateRangePicker component's accessibility levels are ensured through an [accessibility-checker](#) and [axe-core](#) software tools during automated testing.

The accessibility compliance of the DateRangePicker component is shown in the following sample. Open the [sample](#) in a new window to evaluate the accessibility of the DateRangePicker component with accessibility tools.

See also

- [Accessibility in Syncfusion React components](#)

Style appearance in React Daterangepicker component

The following content provides the exact CSS structure that can be used to modify the control's appearance based on the user preference.

Customizing the appearance of DateRangePicker wrapper element

Use the following CSS to customize the appearance like height and font size of the wrapper element.

```
`css
```

```
/ To specify height and font size /
```

```
.e-input-group input.e-input, .e-input-group.e-control-wrapper input.e-input {
```

```
font-size: 20px;
```

```
height: 40px;
```

```
}  
`
```

Customizing the DateRangePicker icon element

Use the following CSS to customize the DateRangePicker icon element

```
`css  
  
/ To specify background color and font size /  
  
.e-input-group .e-input-group-icon:last-child, .e-input-group.e-control-wrapper .e-input-group-icon:last-child {  
  
background-color: darkgray;  
  
font-size: 14px;  
  
}  
`
```

Customizing the DateRangePicker popup calendar header

Use the following CSS to customize the DateRangePicker popup calendar header

```
`css  
  
/ To specify background and height /  
  
.e-daterangepicker.e-popup .e-range-header {  
  
background: beige;  
  
height: 80px;  
  
}  
`
```

Customizing the DateRangePicker popup calendar header title

Use the following CSS to customize the DateRangePicker popup calendar header title

```
`css  
  
/ To specify color and font size /  
  
.e-daterangepicker.e-popup .e-range-header .e-start-label, .e-daterangepicker.e-popup .e-range-header .e-end-label {  
  
color: brown;  
  
font-size: 30px;  
  
}  
`
```

Customizing the DateRangePicker popup calendar content

Use the following CSS to customize the DateRangePicker popup calendar content

```
`css  
  
/ To specify background color /
```

```
.e-daterangepicker.e-popup .e-calendar {  
background-color: brown;  
}  
`css
```

Customizing the DateRangePicker popup calendar content title

Use the following CSS to customize the DateRangePicker popup calendar content title

```
`css  
  
/ To specify color and font size /  
  
.e-daterangepicker.e-popup .e-calendar .e-header .e-title {  
color: beige;  
font-size: 20px;  
}  
`css
```

Customizing the DateRangePicker popup calendar previous and next icon

Use the following CSS to customize the DateRangePicker popup calendar previous and next icon

```
`css  
  
/ To specify font size /  
  
.e-calendar .e-header .e-prev, .e-calendar .e-header .e-next, .e-bigger.e-small .e-calendar .e-header .e-prev, .e-bigger.e-small .e-calendar .e-header .e-next {  
font-size: 20px;  
}  
`css
```

Customizing the DateRangePicker popup calendar date cell grid on hovering

Use the following CSS to customize the DateRangePicker popup calendar date cell grid on hovering

```
`css  
  
/ To specify background color and border /  
  
.e-calendar .e-content td:hover span.e-day {  
background-color: beige;  
border: 1px solid black;  
}  
`css
```

Customizing the DateRangePicker popup calendar primary button in footer

Use the following CSS to customize the DateRangePicker popup calendar primary button in footer

```
`css
```

/ To specify background color and border color /

```
.e-daterangepicker.e-popup .e-footer .e-btn.e-apply.e-flat.e-primary:disabled, .e-daterangepicker.e-  
popup .e-footer .e-btn.e-apply.e-flat.e-primary:disabled, .e-daterangepicker.e-popup .e-footer .e-css.e-  
btn.e-apply.e-flat.e-primary:disabled, .e-daterangepicker.e-popup .e-footer .e-css.e-btn.e-apply.e-flat.e-  
primary:disabled {
```

```
background-color: brown;
```

```
border-color: black;
```

```
}
```

```
,
```

[Customizing the DateRangePicker popup calendar cancel button in footer](#)

Use the following CSS to customize the DateRangePicker popup calendar cancel button in footer

```
`css
```

/ To specify background color, color, and border color /

```
.e-daterangepicker.e-popup .e-footer .e-btn.e-flat, .e-daterangepicker.e-popup .e-footer .e-css.e-btn.e-  
flat {
```

```
background-color: beige;
```

```
border-color: black;
```

```
color: maroon;
```

```
}
```

```
,
```

[Customizing the footer element in the DateRangePicker popup calendar](#)

Use the following CSS to customize the DateRangePicker popup calendar footer element

```
`css
```

/ To specify background color, color, and border color /

```
.e-daterangepicker.e-popup .e-footer {
```

```
background-color: beige;
```

```
height: 50px;
```

```
}
```

```
,
```

[Customizing the selected date cell grid in the DateRangePicker popup calendar](#)

Use the following CSS to customize the selected date cell grid in the DateRangePicker popup calendar

```
`css
```

/ To specify background and border /

```
.e-calendar .e-content td.e-focused-date.e-today span.e-day {
```

```
background: lightgrey;
```

```
border: 1px solid black;
```

```
}
```

```
,
```

Full screen mode support in mobiles and tablets

The DateRangePicker component's full-screen mode feature enables users to view the component popup element in full-screen mode on mobile devices with improved visibility and a better user experience. It is important to mention that this feature is exclusively available for mobile and tablet devices in both landscape and portrait orientations. To activate the full screen mode within the DateRangePicker component, simply set the [fullScreenMode](#) API value to `true`. This action will extend the calendar and presets popup element to occupy the entire screen on mobile devices.

```
`typescript
```

```
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
```

```
import * as React from "react";
```

```
import * as ReactDOM from "react-dom";
```

```
export default class App extends React.Component<{}, {}> {
```

```
  private mobileMode:boolean = true;
```

```
  public render() {
```

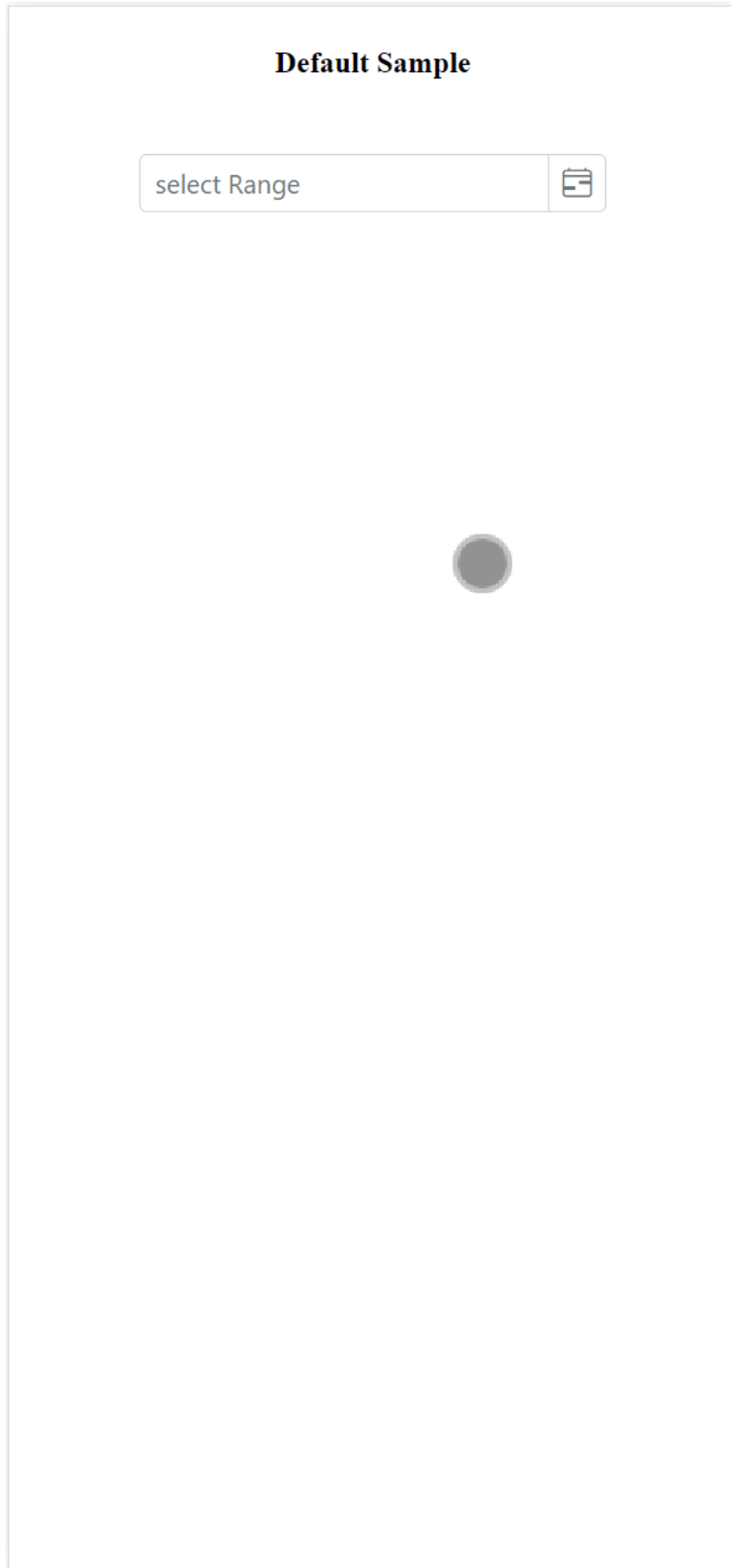
```
    return <DateRangePickerComponent id="daterangepicker" fullScreenMode={this.mobileMode} />
```

```
  }
```

```
};
```

```
ReactDOM.render(<App />, document.getElementById('element'));
```

```
,
```



![DateRangePickerPresetsFullScreen](../images/DateRangePickerrPresetsFullScreen.gif)

How To

Disable the `daterangepicker` component in React `Daterangepicker` component

`DateRangePicker` can be inactivated on a page, by setting `enabled` value as `false` which will disable the component completely from all user interactions including in the form post. The following example demonstrate the disabled component.

INDEX.JSX

```
// import the datepickercomponent
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
  disable = false;
  render() {
    return <DateRangePickerComponent placeholder='Select a range'
    enabled={this.disable}/>;
  }
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
  private disable:boolean = false;
  public render() {
    return <DateRangePickerComponent placeholder='Select a range'
    enabled={this.disable} />
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

Customize the `daterangepicker` day header in React `Daterangepicker` component

You can change the format of the day that to be displayed in header using `dayHeaderFormat` property. By default, the format is `Short`.

You can find the possible formats on below.

Name	Description
Short	Sets the short format of day name (like Su) in day header.
Narrow	Sets the single character of day name (like S) in day header.
Abbreviated	Sets the min format of day name (like Sun) in day header.
Wide	Sets the long format of day name (like Sunday) in day header.

INDEX.JSX

```
{% raw %}
// import the daterangepickercomponent
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
  daterangepickerObj;
  floatLabelObj;
  floatData;
  fields;
  value = 'Short';
  constructor(props) {
    super(props);
    this.floatData = [
      { Id: 'Short', Label: 'Short' },
      { Id: 'Narrow', Label: 'Narrow' },
      { Id: 'Abbreviated', Label: 'Abbreviated' },
      { Id: 'Wide', Label: 'Wide' }
    ];
    this.fields = { text: 'Label', value: 'Id' };
  }
  formatHandler(args) {
    this.daterangepickerObj.dayHeaderFormat = args.value;
  }
  render() {
    return (<div id='container'>
      <div id='daterangepicker'>
        <DateRangePickerComponent ref={ (daterangepicker) =>
this.daterangepickerObj = daterangepicker} cssClass="format-wide"
dayHeaderFormat="Short"/>
      </div>
      <div id="format">
        <label className="custom-input-label">Header Format
Types</label>
        <DropDownListComponent id="select" value={this.value}
dataSource={this.floatData} ref={ (dropdownlist) => { this.floatLabelObj =
dropdownlist; }} fields={this.fields}
change={this.formatHandler.bind(this)} />
      </div>
    </div>);
  }
}
;
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}
```

INDEX.TSX

```
{% raw %}
// import the daterangepickercomponent
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import { DropDownListComponent } from '@syncfusion/ej2-react-dropdowns';
import * as React from "react";
```

```

import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
    public daterangepickerObj: DateRangePickerComponent;
    public floatLabelObj: DropDownListComponent;
    private floatData: { [key: string]: Object }[];
    private fields: object;
    private value: string = 'Short';
    constructor(props: {}) {
        super(props);
        this.floatData = [
            { Id: 'Short', Label: 'Short' },
            { Id: 'Narrow', Label: 'Narrow' },
            { Id: 'Abbreviated', Label: 'Abbreviated' },
            { Id: 'Wide', Label: 'Wide' }
        ];
        this.fields = { text: 'Label', value: 'Id' };
    }

    private formatHandler(args: any): void {
        this.daterangepickerObj.dayHeaderFormat = args.value;
    }

    public render(): JSX.Element {
        return (
            <div id='container'>
                <div id='daterangepicker'>
                    <DateRangePickerComponent ref={(daterangepicker) =>
this.daterangepickerObj = daterangepicker} cssClass="format-wide"
dayHeaderFormat="Short"/>
                </div>
                <div id="format">
                    <label className="custom-input-label">Header Format
Types</label>
                    <DropDownListComponent id="select" value = {this.value}
dataSource={this.floatData} ref={(dropdownlist) => { this.floatLabelObj =
dropdownlist }} fields={this.fields} change={this.formatHandler.bind(this)}
/>
                </div>
            </div>
        );
    }
};
ReactDOM.render(<App />, document.getElementById('element'));
{% endraw %}

```

Set the placeholder in React Daterangepicker component

The following example demonstrates how to set [placeholder](#) in the DateRangePicker control.

Using `placeholder` you can display a short hint in the input element.

INDEX.JSX

```

// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
    render() {

```

```

        return <DateRangePickerComponent placeholder='Select a range' />;
    }
}
;
ReactDOM.render(<App />, document.getElementById('element'));

```

INDEX.TSX

```

// import the daterangepicker component
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
    public render() {
        return <DateRangePickerComponent placeholder='Select a range' />
    }
};
ReactDOM.render(<App />, document.getElementById('element'));

```

Customization using cssclass in React Daterangepicker component

To customize UI, you can make use of [cssClass](#) which will be added to DateRangePicker component as the root CSS class. With this CSS class, you can override existing styles of DateRangePicker.

Following is the list of classes that provides flexible way to customize the DateRangePicker component.

Class Name	Description
---	---
e-date-range-wrapper	Applied to DateRangePicker wrapper
e-range-icon	Applied to the DateRangePicker icon.
e-popup	Applied to DateRangePicker popup wrapper.
e-calendar	Applied to both Calendar element.
e-right-calendar	Applied to right Calendar element.
e-left-calendar	Applied to left Calendar element.
e-start-label	Applied to start label in popup.
e-end-calendar	Applied to end label in a popup.
e-day-span	Applied to day span details label in a popup.
e-footer	Applied to footer elements in a popup.
e-apply	Applied to apply button in footer in a popup.
e-cancel	Applied to cancel button in footer in a popup.
e-header	Applied to Calendar header.
e-title	Applied to Calendar title.
e-icon-container	Applied to Calendar previous and next icon container.
e-prev	Applied to Calendar previous icon.

- | e-next | Applied to Calendar next icon.|
- | e-weekend | Applied to Calendar weekend dates.|
- | e-other-month | Applied to Calendar other month dates.|
- | e-day | Applied to each day cell of the Calendar.|
- | e-selected | Applied to Calendar selected dates.|
- | e-disabled | Applied to Calendar disabled dates.|

In the following example, we have customized the DateRangePicker color of texts, background of selected text using `cssClass`.

INDEX.JSX

```
// import the datepickercomponent
import { DateRangePickerComponent } from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component {
  render() {
    return <DateRangePickerComponent cssClass='customCSS'
placeholder='Select a range' />;
  }
}
;
ReactDOM.render(<App />, document.getElementById('element'));
```

INDEX.TSX

```
// import the datepickercomponent
import { DateRangePickerComponent} from '@syncfusion/ej2-react-calendars';
import * as React from "react";
import * as ReactDOM from "react-dom";
export default class App extends React.Component<{}, {}> {
  public render() {
    return <DateRangePickerComponent cssClass='customCSS'
placeholder='Select a range' />
  }
};
ReactDOM.render(<App />, document.getElementById('element'));
```

Ej1 api migration in React Daterangepicker component

This article describes the API migration process of DateRangePicker component from Essential JS 1 to Essential JS 2.

Date Selection

```
{% raw %}
```

```
<!-- markdownlint-disable MD033 -->
```

Behavior	API in Essential JS 1	API in Essential JS 2
Setting the value	Property: value	Property: value <code>this.value = [new Date('1/15/2017'), new Date("1/15/2017")];</code>

Date Format

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Display the date format	Property: dateFormat	Property: format

Date Range

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Minimum date	Property: minDate	Property: min
Maximum date	Property: maxDate`	Property: max
Start date	Property: startDate	Property: startDate
End date	Property: endDate	Property: endDate`

Disabled Dates

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Disabled dates	Not Applicable	Can be achieved by <code>disableDaterange(args) { if (args.date.getDay() === 0 args.date.getDay() === 6) { args.isDisabled = true; }}</code>

Customization

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
CSS Class	Property: cssClass	Property: cssClass
Enable/Disable TimePicker with DateRangePicker	Property: enableTimePicker	Not Applicable
Time format	Property: timeFormat	Not Applicable
Minimum days span	Not Applicable	Property: minDays
Maximum days span	Not Applicable	Property: maxDays
Button text	Property: buttonText buttonText = { reset: "Again", cancel: "Cut", apply: "Ok" }	Can be achieved by <code>L10n.load({ 'en': { 'daterangepicker': { applyText: 'Apply' } } });</code>
Show/Hide the popup button	Property: showPopupButton	Event: focusonFocus(args) { this.show(); }.e-control-wrapper.e-input-group-icon.e-range-icon { display: none; }
Enable/Disable the rounded corner	Property: showRoundedCorner	Can be achieved by <code>.e-control-wrapper.e-custom-style.e-date-range-wrapper.e-input-group { border-radius: 4px; }</code>
FocusIn event	Not Applicable	Event: focusonFocus(args) { / code block */ }
FocusOut event	Not Applicable	Event: bluronBlur(args) { / code block */ }
FocusIn method	Not Applicable	Method: focusIn()onCreate(args) { this.focusIn(); }
FocusOut method	Not Applicable	Method: focusOut()onCreate(args) { this.focusOut(); }

Accessibility

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Enable/Disable the RTL	Not Applicable	Property: enableRtl

Persistence

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Enable/Disable the persistence	Property: enablePersistence	Property: enablePersistence

Common

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Width	Property: width	Property: width
Readonly	Not Applicable	Property: readonly
Show/Hide the clear button	Not Applicable	Property: showClearButton
Height	Property: height	Can be achieved by.e-control-wrapper.e-custom-style.e-date-range-wrapper.e-input-group { height: 35px;}
Show/Hide the week number	Not Applicable	Property: weekNumber
Watermark text	Property: watermarkText	Property: placeholder
Enable/Disable	Property: enabled	Property: enabled
Disable the DateRangePicker	Method: disable() function onCreate() { var daterangeObj = \$("#daterangepicker").data("ejDateRangePicker"); daterangeObj.disable();}	Not Applicable
Enable the DateRangePicker	Method: enable() function onCreate() { var daterangeObj = \$("#daterangepicker").data("ejDateRangePicker"); daterangeObj.enable();}	Not Applicable
Enable/Disable the input textbox editing	Property: allowEdit	Property: allowEdit
Specify the placeholder text behavior	Not Applicable	Property: floatLabelType
zIndex	Not Applicable	Property: zIndex
Separator	Property: separator	Property: separator

Globalization

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Locale	Property: locale	Property: locale
First day of week	Not Applicable	Property: firstDayOfWeek

Strict mode

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Strict mode	Not Applicable	Property: strictMode

Open and Close

<!-- markdownlint-disable MD033 -->

Behavior	API in Essential JS 1	API in Essential JS 2
Close	Event: closefunction onClose() {/ code block */}	Event: closeonClose(args) {/ code block */}
Hide	Method: popupHide()function onCreate(args) { var daterangeObject = \$("#daterangepicker").data("ejDateRangePicker"); daterangeObject.popupShow(); daterangeObject.popupHide();}	Method: hide()create (args) { this.show(); this.hide();}
Open	Event: openfunction onOpen(args) {/ code block */ }	Event: openonOpen(args) {/ code block */ }
Show	Method: popupShow()function onCreate(args) { var daterangeObject = \$("#daterangepicker").data("ejDateRangePicker"); daterangeObject.popupShow();}	Method: show()create (args) { this.show();}

{% endraw %}